SOME NEW LOCALIZED QUALITY OF SERVICE MODELS AND ALGORITHMS FOR COMMUNICATION NETWORKS

The development and evaluation of new localized quality of service routing algorithms and path selection methods for both flat and hierarchical communication networks

Elmabrook B. M. Mustafa

Submitted for the degree of Doctor of Philosophy

Department of Computing

School of Informatics

University of Bradford

Elmabrook B. M. Mustafa

Title: SOME NEW LOCALIZED QUALITY OF SERVICE MODELS AND ALOGORITHMS FOR COMMUNICATION NETWORKS

Keywords: Localized routing, quality of service, bandwidth, delay, path selection, communication networks

Abstract

The Quality of Service (QoS) routing approach is gaining an increasing interest in the Internet community due to the new emerging Internet applications such as real-time multimedia applications. These applications require better levels of quality of services than those supported by best effort networks. Therefore providing such services is crucial to many real time and multimedia applications which have strict quality of service requirements regarding bandwidth and timeliness of delivery.

QoS routing is a major component in any QoS architecture and thus has been studied extensively in the literature. Scalability is considered one of the major issues in designing efficient QoS routing algorithms due to the high cost of QoS routing both in terms of computational effort and communication overhead.
Localized quality of service routing is a promising approach to overcome the scalability problem of the conventional quality of service routing approach. The localized quality of service approach eliminates the communication overhead because it does not need the global network state information.

The main aim of this thesis is to contribute towards the localised routing area by proposing and developing some new models and algorithms. Toward this goal we make the following major contributions. First, a scalable and efficient QoS

Keywords: Localized routing, quality of service, bandwidth, delay, path selection, communication networks

routing algorithm based on a localised approach to QoS routing has been developed and evaluated. Second, we have developed a path selection technique that can be used with existing localized QoS routing algorithms to enhance their scalability and performance. Third, a scalable and efficient hierarchical QoS routing algorithm based on a localised approach to QoS routing has been developed and evaluated.

Acknowledgment

First of all, I thank Allah for his unlimited grace and help to complete this thesis. I would like to express my deep thanks and appreciation to my parents who brought me up and make me realize the importance of education.

I deeply thank my wife and children for their support and sacrifices they made for me and all what they have done in order to provide peace and quit environment.

I thank my supervisor for his valuable support and guidance throughout my research. I am grateful to him for accepting me as a research student and for being patient with me during my stay at the University of Bradford. He has been both a friend and a mentor to me and I consider myself fortunate to have him as my supervisor.

My acknowledgement would not be completed without expressing true regards and admiration to all the friends and colleagues at University of Bradford who constantly provide emotional support and took care of me in many aspects.

Contents

List of figures

Chapter 1

Introduction

1.1 Background

In computer networks such as the Internet, routing is the process of moving data packets from source node to destination node. Routing protocols are responsible for selecting the most appropriate path to destination. This process is carried out in two steps. The first step is the collection of network state information and the second step is the selection of the best path between source node and destination node based on the collected network state. Routed protocols are used for delivering data packets from source node to destination node along the selected path.

Originally, routing in the Internet was implemented using a best effort method that was concerned with connectivity. Routing protocols such as OSPF [1] that

employs the best effort method usually consider a single metric to select the best available path. Then, use this path to deliver data packets from source node to destination node. For example, delay and shortest path algorithms are used for best path computation. The data packets are forwarded as quickly as possible, but there is no guarantee as to timeliness or actual delivery. The network itself does not actively differentiate between types of traffic carried over the network. In best effort networks, all packets are treated in the same fashion. The network does its best to deliver every packet as quickly as it can, but makes no attempt to treat any class of packets preferentially to any other. This approach is known as best-effort routing and the Internet was originally designed based on this approach.

The Quality of Service approach, abbreviated as QoS, is a relatively new paradigm which is gaining an increasing interest in the Internet community due to the new emerging applications such as real-time multimedia applications that require quality of services higher than the services offered by the best effort networks. Hence providing such services is essential to many real time and multimedia applications which have strict requirements regarding bandwidth and timeliness of delivery. Constraint Based Routing (CBR) [2], Traffic Engineering (TE) [3], Resource Reservation Protocol (RVSP) [4], Multi-Protocol Label Switching (MPLS) [5, 6], Integrated Services (IntServ) [7] and Differentiated Services (DiffServ) [8] are six key QoS architectures and mechanisms suggested to provision QoS multimedia applications.

In QoS routing, paths for delivering data packets from source to destination are selected based upon the knowledge of the QoS state (Network resource availability) at each node in the network and the QoS requirements of the traffic. It is expected that QoS routing will select a path that satisfies QoS requirements of the traffic. Therefore, QoS routing can extensively improve the network performance due its knowledge of the network resource availability.

QoS routing schemes need a knowledge of the global network state in order to take a routing decision. This knowledge of the network state can be obtained through periodic information exchange among routers in a network and this global view of the network is stored in each router of a network. Finally, routers will use their global view of the network to select the optimal path for network traffic. Under this scheme, a router will perform well when it has an accurate global view of the network or network state. However, when the network state changes due to network dynamics such as changing of bandwidth availability, maintaining a precise network state is unrealistic due to prohibitive communication and processing overheads entailed by frequent network state information exchange. In the presence of an inaccurate network state, the QoS schemes may suffer degrading performance as well as potential instability.

1.2 Motivation

Routing in computer networks is based on two tasks, the first task is to find the best route between two nodes. To find this route, routing ptotocols collect the

state information of the network and keep it up-to-date. This provides each node with a consistent view of the network at some instance. Then, baesd on collected network state routing protocols find the most efficient path for network traffic. There are inherent scalability issues for both tasks. As for the first one, collecting the global network QoS state gives raise to several problems:

- Every node in the network must have a global view of the network. This requires that the state information at every source node has to be updated frequently enough to reflect the flow dynamics, which imposes a large communication overhead.

- Due to this overhead and the non-negligible propagation delay of state messages, the link-state algorithm can only provide an inaccurate global QoS state. This inaccuracy has a significant impact on the performance of routing algorithms.

- Out-of-date state information due to large update intervals can cause route instability in global QoS routing algorithms. That is, when the utilization on a link is low, an update causes all the source nodes to prefer routes along this path, resulting in a rapid increase in its utilization. Similarly when the utilization is high, an update causes all the sources to refrain from using this link and consequently its utilization decreases rapidly.

All these problems get worse as the size of the network increases and so the scalability of QoS routing algorithms becomes a major challenge. As for the second task, the computational overhead for computing a feasible path based on the global network QoS state is very high considering that QoS routing is typically done on a per-connection basis.

These challenges motivate the investigation of methods and techniques for enhancing the inherent scalability of QoS routing algorithms in the context of the two tasks described above.

A feasible QoS routing scheme is localized QoS routing as proposed by [9] and [10]. The aim of this scheme is to overcome the problems of conventional QoS routing schemes. Under this scheme, instead of periodically exchanging the network state with other nodes to obtain a global view of the network state, a source node infers the network QoS state from locally collected flow statistcs. This thesis mainly focuses on localized QoS routing schemes and ways to improve their performance.

1.3 Research Aims and Objectives

The aim of this thesis is to contribute towards enhancing the inherent scalability of QoS routing algorithms. This aim can be divided into the following sub-aims:

1. To develop a Dynamic Path Substitution Localized QoS Routing scheme using a dynamic path selection method with bandwidth as a single metric.
2. To develop a Dynamic Path Substitution Localized QoS Routing scheme using the dynamic path selection method using Delay as a single metric.
3. To develop a Disjoint Path Localized QoS Routing scheme using bandwidth as a single metric.
4. To develop a Disjoint Path Localized QoS Routing scheme using delay as a single metric.
5. To develop a Hierarchical Localized QoS Routing Scheme based on a localized QoS approach to QoS routing.

These aims are to be achieved through the following objectives:

- To identify through a comprehensive literature review of contemporary approaches for QoS routing and their related scalability problems.
- To develop an efficient Dynamic Path Substitution Localized QoS Routing scheme.
- To develop an efficient Disjoint Path Localized QoS Routing scheme.

- To develop a simulation environment that will be used to assess the performance of the Dynamic Path Substitution Localized QoS Routing scheme and Disjoint Path Localized QoS Routing scheme.

- To study their performance compared with other localized QoS routing algorithms using bandwidth and delay.

- To develop a Hierarchical Localized QoS Routing algorithm based on a localised approach to QoS routing.

1.4 Thesis Original Contributions

The contribution of this thesis can be stated as below:

- The development of Dynamic Path Substitution Localized QoS Routing scheme based on a localised approach to QoS routing. This scheme creates three candidate sets: Candidate Path set, alternative Candidate Path set and Reserve Candidate path set. Then it replaces saturated paths from the first two sets with a path from the third set. We found that the substitution of a saturated path improves the performance of localized QoS Routing schemes.

- The development of the Disjoint Path Localized QoS Routing scheme based on a localised approach to QoS routing. In this scheme we choose paths that have the fewest common links when creating the candidate path set and alternative candidate path set. Simulation studies showed that this

method improves the performance of localised QoS routing schemes dramatically.

- The development of Hierarchical Localized QoS Routing scheme based on a localised approach to QoS routing. This scheme is developed by implementing the Disjoint Path Localized QoS Routing scheme in the sub network level and in the aggregated level.

1.5 Outline of the thesis

The rest of the thesis is organised as follows.

Chapter 2 describes the notion of QoS and its background. In particular, we describe the issues involved in QoS-based routing and discuss the various network QoS schemes and strategies. The advantages and limitations of different QoS routing schemes and strategies are also discussed and compared based on source routing, distributed routing.

Chapter 3 gives an overview and defines the localized QoS routing schemes. The need of this approach is justified. A number of localised routing algorithms are discussed and their advantages and disadvtages presented.

Chapter 4 presents the simulation environment that we have used. Simulation methodologies, graph model, simulation parameter settings and performance measures are included.

Chapter 5 introduces and describes a novel Dynamic Path Substitution Localized QoS Routing scheme and evaluates its performance compared to the credit based routing algorithm. The Dynamic Path Substitution Localized QoS Routing scheme is implemented using bandwidth as a single metric and delay as a single metric.

Chapter 6 introduces and describes a new Disjoint Path Localized QoS Routing scheme and gives a complete description of its performance compared to the credit based routing algorithm. The Disjoint Path Localized QoS Routing scheme is implemented using bandwidth as a single metric and delay as a single metric.

Chapter 7 introduces a new Hierarchical Localized QoS Routing scheme and presents its simulation results. The Hierarchical Localized QoS Routing scheme is implemented using bandwidth as a single metric.

Chapter 8 concludes the thesis and suggests possible future directions for localised QoS routing schemes.

Chapter 2:

Quality of Service Routing

2.1 Traditional Network Routing

Traditional network routing is mainly concerned with connectivity. Routing protocols usually characterize the network with a single metric, such as hop count or delay, and use shortest-path algorithms for path computation. As a result, routing decisions are made without any awareness of resource availability and requirements. This means that flows are often routed over paths that are unable to support their requirements, while alternate paths with sufficient resources are available. This may result in significant deterioration in performance in terms of call blocking probability.

2.2 Quality of Service Routing

Quality of Service Routing (QoS) aims to meet the QoS requirements of the applications, such as video conferences, and improve network performance. Therefore, strict resource constraints may have to be imposed on the paths being used. QoS routing is the process of selecting the path to be used by the network traffic based on the network traffic QoS requirements, such as bandwidth or delay. It refers to a set of routing algorithms that are able to identify a path that has sufficient residual (unused) resources to satisfy the QoS constraints of a given network traffic [11-20]. Such a path is called a feasible path. In addition, most QoS routing algorithms also consider the optimization of resource utilization measured by metrics such as bandwidth or delay.

2.3 Quality of Service Routing Metrics

Quality of service routing Metrics are parameters used by quality of service routing protocols to determine a path that satisfies the requirements of network traffic. Therefore, very careful attention must be given when using dynamic metrics that affect routing decisions in order to preserve routing stability. The metrics have to be selected so that the requirements can be presented by one metric or a reasonable combination of them. As the metrics define the types of QoS guarantees the network is able to support, no requirement can be supported if it cannot be mapped onto a combination of

the selected metrics [21]. The metrics commonly used on QoS routing and constraint-based routing are divided into three main categories, also called the composition rules of the metrics [18].

2.3.1 Additive Metrics

The rule for an additive metrics composition is that the value of the metrics over a path is the sum of the values over each link. Equation (1) represents the total metric value $w(P)$ which is the sum of links metrics $w(u_1, u_2), w(u_3, u_4), ..., w(u_{i-1}, u_i)$ over path $P = ((u_1, u_2), (u_{3,}, u_4), ..., (u_{i-1}, u_{i,}))$.

$$w(P) = w(u_1, u_2) + w(u_3, u_4) + ... + w(u_{i-1}, u_i) \quad \text{..... (1)}$$

Delay, delay jitter and number of hops are examples of additive metrics.

2.3.2 Multiplicative Metrics

Multiplicative Metrics can be formed as shown in equation (2). In this equation the total metric value $w(P)$ is the product of link metrics $w(u_1, u_2), w(u_3, u_4), ..., w(u_{i-1}, u_i)$ over path $P = ((u_1, u_2), (u_{3,}, u_4), ..., (u_{i-1}, u_{i,}))$.

$$w(P) = w(u_1, u_2) * w(u_3, u_4) * ... * w(u_{i-1}, u_i) \quad \text{..... (2)}$$

Loss probability is common example of a multiplicative metric.

2.3.3 Concave Metrics

The value of a concave metric $w(P)$ over a path $P = ((u_1, u_2), (u_{3,}, u_4), ..., (u_{i-1}, u_{i,}))$ corresponds to the minimum value observed in

all links of that path with link metrics $w(u_1, u_2), w(u_3, u_4), ..., w(u_{i-1}, u_i)$ as shown by equation (3).

$$w(P) = \min(w(u_1, u_2), w(u_3, u_4), ..., w(u_{i-1}, u_i) \quad \text{..... (3)}$$

Bandwidth is a common example of a concave metric. The bandwidth is the residual bandwidth that is available for new network traffic. It can be defined as the minimum of the residual bandwidth taken over all links on the path, also called the bottleneck bandwidth.

2.4 Quality of Service Routing Types

Quality of service routing can be classified according to many different criteria.

2.4.1 Routing Information Source

This type of routing is related to the source of the network state information. The following describes the types of network state information collection methods:

2.4.1.1 Local State Information

In order to make routing decisions a node in a network has to own up to date network state information [14]. In this of routing, a node has to obtain

the network state information locally. It maintains a routing table that contains information about all directly reachable nodes by that node.

2.4.1.2 Global State Information

The global state information is the collection of local state of all nodes in a network. Every node in a network has to maintain the global state by using a link-state protocol [22] or by using a distance vector protocol [23]. These protocols exchange the network state among network nodes. The link-state protocols broadcast the local state of every node to every other node so that each node knows the topology of the network. The distance-vector protocols periodically exchange distance vectors among adjacent nodes. A distance vector has an entry for every possible destination, consisting of the property of the best path and the next node on the best path.

2.4.1.2.1 Link State Routing

Link state routing protocols or shortest path first protocols such as OSPF [22] flood network state information to all nodes in the network. Each node sends network state information that describes the latest state of its links.

2.4.1.2.2 Distance Vector Routing

Distance vector routing protocols or Bellman-Ford protocols require every node to send part of or all its network state information to its neighbours. For example RIP [23] uses the hop count to a destination node whereas IGRP [24] takes into consideration other network state information such as delay and available bandwidth. When a node receives the updated network

state from its neighbours it can adjust its own view of the network accordingly to reflect the changes and then inform its neighbours of the changes.

2.4.2 Number of Sources and Destinations

Routing protocols can be classified according to the number of nodes that participate in the routing event. When there is a single node as a sender and a single node as a recipient, the routing event is called unicast routing. In case of one node a sender and a group of nodes as receivers, the routing event is called multicast routing. Flood routing or broadcast routing is the routing event in which one node acts as a sender and all other nodes in the networks act as receivers.

2.4.2.1 Unicast Routing

Unicast routing is the process for forwarding network traffic from a node to another node in the network. The network traffic is addressed to a unique address. Therefore, the network traffic contains the address of the destination node as shown in figure 2.1. Node A forwards network traffic to node F through intermediate nodes B and D.

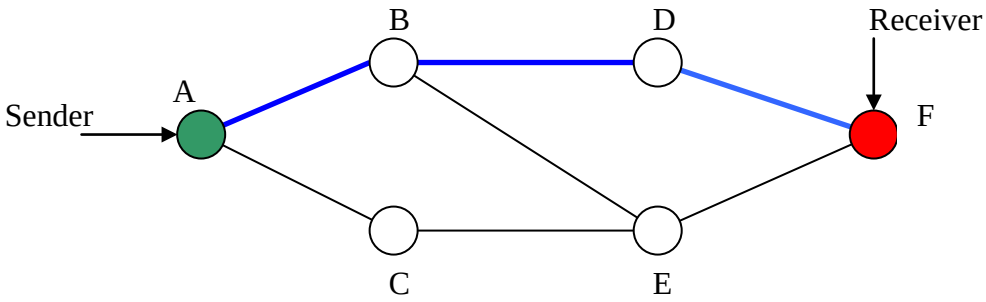


Figure 2.1: Unicast Routing

2.4.2.2 Multicast Routing

Multicast routing enables a node to send network traffic to a subset of the other network nodes. Multicast routing is carried out in two steps. The first step is to find a tree that connects all nodes in a multicast group and then send the network traffic to this group, as shown in figure 2.2. The source node A transmits network traffic to a subset of network nodes {B, E, D}. The sender and the group of receivers form a multicast group which have a single identifier (address).

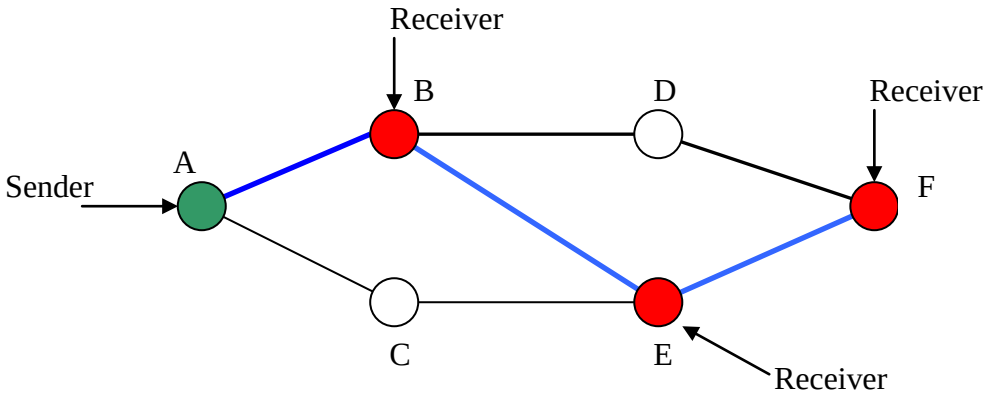


Figure 2.2 : Multicast routing

2.4.2.3 Flood Routing

In flood routing or broadcast routing a source node sends network traffic to all other nodes in the network, as shown in figure 2.3. The source node A broadcasts network traffic to all other nodes in the network.

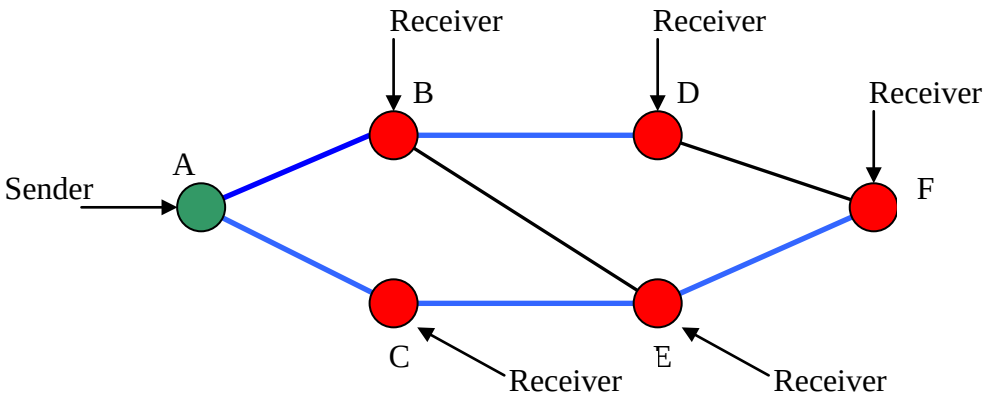


Figure 2.3: Flood Routing

2.4.3 Routing Decision Location

A routing decision is the process of finding a feasible path from source to destination. This decision can be taken in the source node or in each intermediate node in the path.

2.4.3.1 Source Routing

Source routing is a type of unicast routing. In this type of routing there is some network traffic that needs to be transmitted from one node, which is known as the source node to another node, which is known as the destination node. The path that is used to transmit network traffic data from source node to destination node is computed or determined by the source node using metrics such as delay or bandwidth. Therefore, the source node should have knowledge of the current state and topology of the entire network that contains both source node and destination node. The computation of the optimal path from source to destination can be

done centrally in the source node by applying a shortest path algorithm such as Dijkstra's Algorithm [25, 26] or the Bellman-Ford Algorithm [26, 27]. If the source node cannot find a efficient path, the source node may reject the connection or negotiate for lower requirements with the originator of the connection. If the source node finds the efficient path, it sends a control message along that optimal path to inform the intermediate nodes about that path and each node takes some well defined actions to setup the connection along the selected path.

Source routing can be classified according to a node's knowledge of the global state of the network into static or dynamic routing. Static routing uses a static topology to select a path but dynamic routing uses the information about the available resources on each link available throughout the network, so that any source can have the most current resource information on each link. Static routing has very low communication overhead but it has the drawback of a poor flow-setup success rate. On the other hand dynamic routing needs to maintain accurate global network state information which becomes very difficult to obtain in large scale networks.

Source routing can be also classified as on-demand computation [14, 28] and pre-computation [29, 30]. In the on-demand computation, a QoS path is computed on a pre-request basis that increases the computational overhead. In contrast, in the per-computation approach QoS paths are computed asynchronously with request arrivals. These paths are used to

establish multiple network traffic that has the same destination thus reducing computational overhead

Although source routing is simple to implement, it has many problems. The first one is the overhead problem. The overhead associated with maintaining and exchanging the complete global state is very high, especially in large networks [14]. Exchanging the global state very frequently enables the routing process to cope with the dynamics of the network state and enhance the accuracy of the routing algorithm but introduces excessively high overhead. Reducing the exchange frequency can help but at the price of reducing the accuracy of the routing algorithm due to using an outdated network state. Hence, we have a trade-off situation and a good balance needs to be achieved. Second, the computational overhead at the source node is very high because all path computations are done in a single node [14]. In summary, source routing suffers from scalability issues and it may be impractical to use source routing in very large networks. The second one is that the packet header is large as it needs to store the full forwarding path. The third one is the initial computation delay during the connection establishment.

2.4.3.2 Distributed Routing

In distributed routing, the path computation is shared among intermediate nodes between source node and destination node. The routing path is determined on a hop-by-hop basis at every node. In some distributed routing algorithms each node needs to maintain the global network state

information which also suffers from the same issues as that of source routing. However, inconsistency in the global network state information can form loops. Distributed routing algorithms that maintain only the local state information are exempt from these issues. Current Internet routing protocols such as RIP (Routing Information Protocol) [23] uses hop-by-hop routing. The routing computation overhead is less as it is distributed among intermediate nodes and there is very little setup delay. Distributed routing protocols are more scalable compared with source routing protocols.

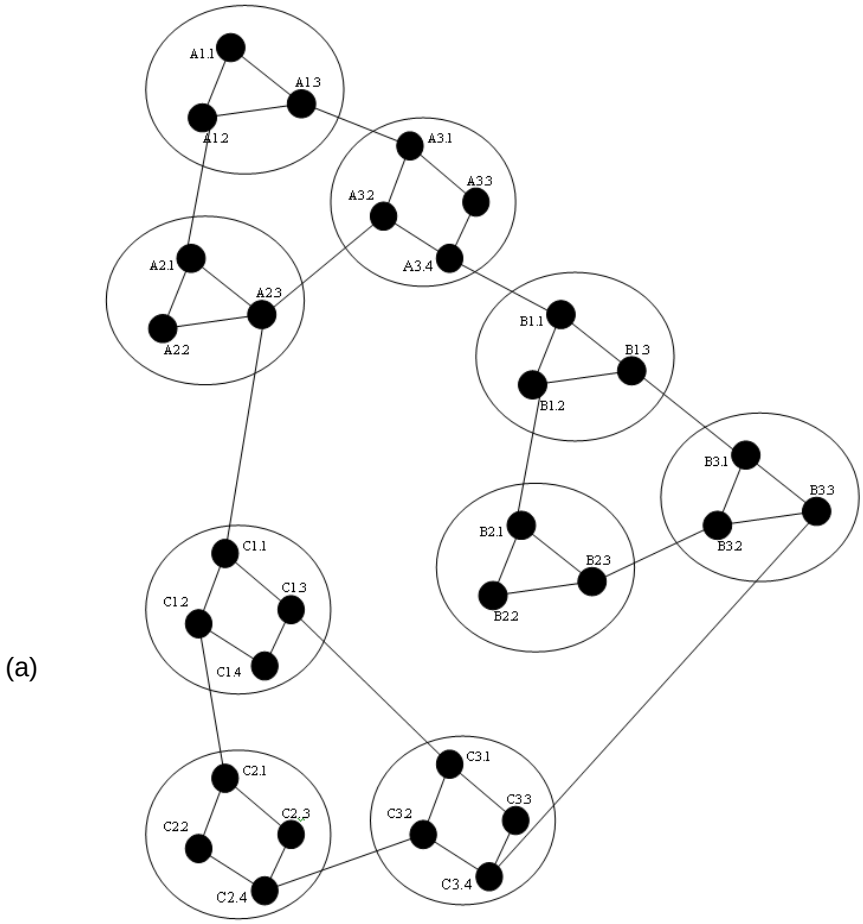
The advantage of distributing routing is the path computations are distributed among many nodes which reduces the routing response time and increases the scalability of the algorithm. It is also possible to search multiple paths in parallel for a feasible one. However, distributed routing needs to deal with the problems associated with distributed algorithms such as loop detection, Algorithms which need global state share some of the drawbacks of source routing, while the algorithms that do not require maintaining the global state usually send more control messages to compensate for the lack of global state.

2.4.3.3 Hierarchical Routing

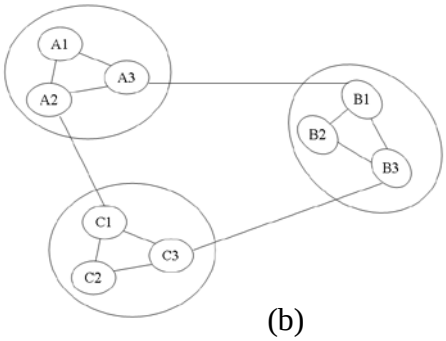
In hierarchical routing, a hierarchical topology is formed by clustering the network nodes into groups that form a logical node. These logical nodes are clustered to new groups to form a higher level logical node and so on. Figure 2.4 shows an actual physical network. The nodes, represented by

the black dots, are clustered into groups of three or four. Figure 2.4(b) shows the resulting aggregated topology, along with the second aggregation, in which the groups are clustered into new groups, in this case consisting of three first level groups.

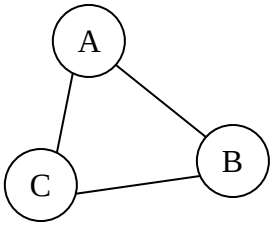
Hierarchical routing proposed for QoS routing [14, 31] is somewhat similar to source routing, but each node has detailed state information only about the nodes in the same group, and aggregated information about nodes in the other groups. Figure 2.4 shows the topology as seen by the node A2.1. Source node computes a path using the aggregated topology. When a node starts to transmit, first a control message is sent. When it reaches a border node which is a part of a group presented as a logical node in the path, it uses source routing to expand the path through the group. A typical example of a hierarchical routing protocol is Private Network-Network Interface, PNNI [32], which has been used for ATM networks.



(a)



(b)



(c)

Figure 2.4 : (a) Clustering in hierarchical network

(b) First level Clustering in hierarchical network

(c) Second level Clustering in hierarchical network

2.5 Quality of Service Architectures

There are many QoS architectures to provision QoS in a computer network such as the Internet. The most well-known are the Constraint Based Routing (CBR) [2], Traffic Engineering (TE) [33], Resource Reservation Protocol (RVSP) [4], Multi-Protocol Label Switching (MPLS) [5, 6], Integrated Services (IntServ) [34] and Differentiated Services (DiffServ) [35].

2.5.1 Constraint Based Routing

Constraint based routing (CBR) represents a type of routing algorithm that computes routing paths based on a set of requirements or constraints. CBR also considers network topology, network traffic requirements and resource availability on links to increase network traffic utilization. The network traffic requirements or constraints can be applied by administrative policies or QoS requirements. Network traffic constraints applied by administrative policies are known as policy constraints. The routing protocols that use this approach are called routing based policies. Network traffic constraints that are applied by QoS requirements, such as delay, bandwidth or loss, are known as QoS constraints. The routing protocols that use this approach are known as QoS routing [14, 21].

Policy routing protocols use paths which match to administrative rules and service level agreements (SLAs). In policy routing, administrators can make routing decisions not only on the destination node location, but also on parameters such as applications used, packet size, or identity of both source and destination end systems.

QoS routing protocols aim to assure multiple QoS requirements or constraints, such as delay and bandwidth boundaries, and also consider the global utilization of network resources [36]. QoS routing protocol performance is dependent on the precision of network state information, the network topology, and the network traffic requirements.

2.5.2 Traffic Engineering

Traffic engineering goals are to develop and improve network performance through optimization of resource utilization in the network. The optimization objective depends on the required service. However, reference [37] lists the following common objectives that include: 1) Minimizing congestion and packet losses in the network. 2) Improving link utilization. 3) Minimizing the total delay experienced by packets. 4) Increasing the number of customers with the current assets.

A traffic engineering system is composed of six main components as shown in figure 2.4: 1)Topology and network state information discovery, 2)Traffic demand estimation, 3)Route computation, 4)Graphical user interface, 5)Network interface, and 6)Data repository.

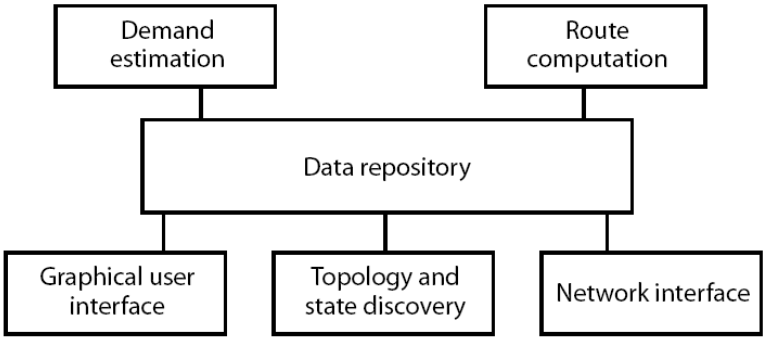


Figure 2.5 : Traffic Engineering System (adapted from [37])

A traffic engineering system should monitor the network topology and network state information and collect any changes that happen to the network due to network dynamics. Network monitoring and collecting network state information is done by the topology and state discovery component.

Network traffic estimation is calculated based on accurate information related to traffic demands of users or calculated based on traffic measurements. The former calculation gets traffic demands from the service agreement between user and service provider. The second calculation gets information from network statistics such as traffic loads on a link and breakdown of traffic types.

Route computation calculates routes based on traffic demands, network state information and a number of constraints. These constraints can be imposed by routing policies or QoS requirements.

After the traffic engineering system has obtained the best possible routes for traffic demands, it has to configure the network elements in the network accordingly. A traffic engineering system can interface a number of options with the network elements.

Topology and state discovery in a traffic engineering system is responsible for maintaining accurate network state information.

2.5.3 Multi-Protocol Label Switching

Multi-Protocol Label Switching (MPLS) MPLS is a high rate packet forwarding scheme where fixed length labels are used to improve the throughput and delay performance of IP networks [38]. It uses standard routing protocols such as OSPF to define paths between sender and receiver, then assigns packets to these paths as they enter the network and uses ATM switches to forward these packets along the paths. The MPLS architecture consists of two parts as shown figure 2.5. Data plane and control plane. The data plane forwards MPLS network traffic based on information provided by the control plane. The data plane contains a Forwarding Information Table (FIL) that contains information about Label Switched Paths (LSPs).

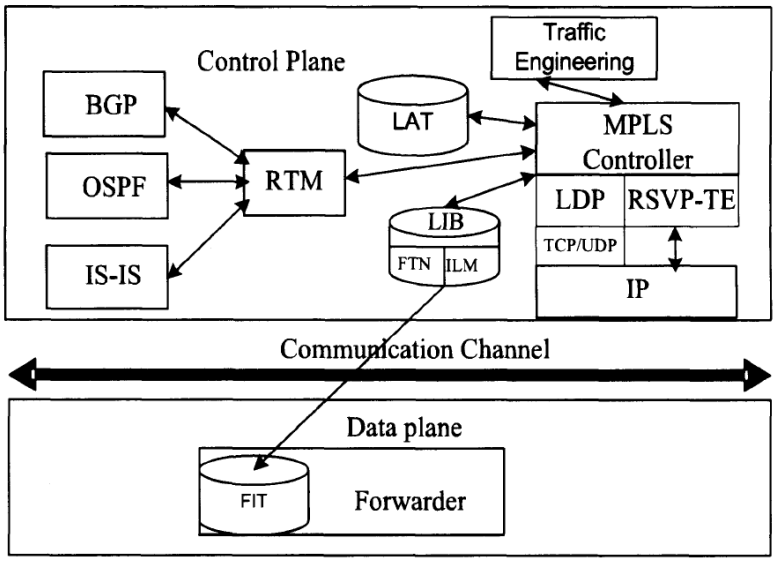


Figure 2.6 : MPLS architecture adapted from [39]

The control plane consists of:

- Signalling protocols. The Label Distribution Protocol (LDP) and Resource ReServation Protocol with Traffic Engineering ability (RSVP-TE) can be used alternatively. They cooperate with Interior Gateway Protocols (IGPs), such as OSPF [7] to compute the paths by Label Edge Routers (LERs) based on traffic engineering criteria.

- Label Allocation Table (LAT). LAT contains the available label space of the router.

- Label Information Base (LIB). LIB is a set of tables containing the label mapping rules. Once a Label Switched Path (LSP) is determined, it is recorded into the FIT located on the data plane.

- Traffic engineering. It acts as a link between the system administrator and the link management module in order to control the bandwidth consumption on LSPs and inform the neighbours.

MPLS improves the performance of an IP network by reducing the amount of per packet processing required at each node in IP networks. MPLS provides new features that ensure its success: QoS support, traffic engineering, virtual private networks and multiprotocol support.

2.5.4 Integrated Services (IntServ)

In Integrated Services (IntServ) [34], network traffic is serviced on a per-flow basis in accordance with each flow's absolute service request. Each network device must reserve resources for each flow and isolate each flow from another. In Intserv, resource reservation protocol (RSVP) signalling protocol is used by a host to request specific qualities of service from the network for particular application data streams or flows. RSVP is also used by routers to deliver quality-of-service (QoS) requests to all nodes along the path(s) of the flows and to establish and maintain state to provide the requested service. RSVP requests will generally result in

resources being reserved in each node along the data path. A flow is either admitted to the network or rejected depending on the availability of enough network resources along the whole path to satisfy its service requirements. An admitted flow is guaranteed the service throughout its holding time on an end-to-end basis. Once a flow of data is admitted to the network, RSVP uses a soft state approach to manage resources in routers and hosts. The RSVP soft state is created and periodically refreshed using RSVP messages to avoid an explicit termination procedure to release resources used by a flow [4].

Intserv has a number of drawbacks that limit its deployment. With the explosive growth of the Internet, the number of flows whose state must be managed at a given instant by each core router is very large (tens of thousands). Isolation of flows in Intserv (having one logical buffer per-flow) may be very expensive when the number of flows is very large, which is normally the case in core routers. The other problem that hinders deployment of Intserv is backward incompatibility; Network devices are required to perform and understand a signalling protocol (RSVP) that is not yet widely deployed. Moreover, the "mice and elephants" paradigm of Internet flows means that most flows are very small. It is obvious that performing RSVP signalling on a per-flow basis for this traffic will be a very complex task. In general, Intserv does not scale particularly well in core routers. The failure of Intserv resulted in another QoS architecture known as differentiated service (Diffserv).

2.5.5 Differentiated Services (DiffServ)

Differentiated Services (DiffServ) is a QoS framework that aggregates individual flows with similar traffic behaviours into a small number of classes and provides relative services among these classes [40]. Diffserv alleviates the complexity in core routers by assigning all complex processing such as network traffic classification and network traffic dimensioning to edge routers. In the backbone, forwarding per-hop behaviours (PHB) are defined, namely, expedited forwarding (EF) and assured-forwarding (AF) [41, 42]. EF-PHB has a higher priority than AF-PHB. EF per-hop behaviour is proposed for traffic that requires zero loss and zero jitter or queueing delay guarantees. EF-PHB is proposed to emulate leased line connections. There are four AF-PHBs each with three drop precedences. Differentiated services are managed along the path by scheduling algorithms and buffer management mechanisms in routers. Priority queuing assures EF traffic is not delayed and buffer mechanisms choose the AF traffic from which to drop packets that are marked to have low priorities during congestion. No admission control or end-to-end complex signalling protocols are required in the Diffserv framework. Instead, in Diffserv, a flow that arrives at the edge router is classified and conditioned according to its traffic contract and is always accepted into the network. Diffserv has a number of drawbacks too despite being scalable and not requiring a signalling protocol. When a single flow in a class violates its traffic contract the services of all other flows of the same class

degrade equally. Diffserv does not provide absolute guarantees and during congestion, a high priority AF class may be demoted to lower priority AF class. Another drawback of Diffserv is that it does not provide end-to-end guarantees, which may be required by some applications. An architecture proposed to enable Diffserv to support end-to-end QoS services is called bandwidth broker (BB) [43]. BB is a centralized network manager that makes it possible to manage network state on a domain basis rather than a router basis. BB has a policy database that tracks user behaviours and allows admission control [44]. BB also maintains service agreements and negotiations among different adjacent domains. The work on BB is still new and its deployment is highly uncertain because it is contrary to the internet philosophy of being a distributed system. Another proposed scalable QoS architecture aimed at supporting end-to-end services is the hybrid of Intserv and Diffserv [45]. The scalability of Diffserv makes it suitable for large networks.In this framework, Intserv and Diffserv are used together to meet the needs of large ISPs who manage the backbones on the Internet and the need of the QoS to end users. In this hybrid framework, it is envisioned that Intserv will be deployed in stub networks and Diffserv in core networks. The RSVP signalling protocol is proposed to reserve resources in stub networks and some Diffserv admission control service (DACS) is proposed at the edge or boundary routers to control admission of flows to the Diffserv domain.

2.6 Summary

In this chapter, quality of service routing has been defined along with its metrics that can be one of a number of types such as additive metrics such as delay and jitter or multiplicative metrics such as loss probability or concave metrics such as bandwidth. After that, QoS routing categories and classes where presented according to routing information source which can be local state or global state, or according to the number of source and destinations such as unicast or multicast routing, or according to routing decision place which can take in source node (source routing) or distributed routing where each node make its own decision. Finally, a number of quality of service architectures were discussed including Constraint Based Routing (CBR), Traffic Engineering (TE), Multi-Protocol Label Switching (MPLS), Integrated Services (IntServ) and Differentiated Services (DiffServ). The main aim of these architectures is to improve the network performance and provide quality of service routing.

Clearly, the main problem with the routing protocols currently deployed in the Internet is the size and the maintenance of an accurate (up-to-date) global link state. This motivates the need for other types of routing where such problems might be avoided.

Chapter 3:

Localized QoS Routing:

3.1 Introduction

The conventional QoS routing algorithms described in chapter 2 depend on global network state information in order to make routing decisions. The global QoS network state information needs to be exchanged periodically among network nodes. This process of exchanging global network state information may introduce large communication and processing overheads which will reduce the performance of the network.

Localized quality of service routing schemes [9, 46-48] are a viable routing scheme that aim to provide better quality of service and overcome the drawbacks of conventional routing schemes by eliminating overheads resulting from periodic exchange of network state information among the network nodes. Localized QoS routing does not need the global network state information to be exchanged among network nodes because it infers the network state information

and eliminates all the problems associated with it. In localized quality of service routing schemes, a source node has to maintain a local view of the network by collecting the network state information locally by monitoring the network traffic generated from itself and transmits this over predetermined sets of candidate paths to each possible destination as required.

3.2 Localized QoS Routing Path Selection

The selection of candidate paths is an important step in localized QoS routing. The method employed to select the set of candidate paths has significant impact on the performance of localized routing algorithms. Reference [49] raises two important questions regarding candidate path selection: 1) how many paths are needed and 2) how to find these paths. Clearly, the number and the quality of the candidate paths selected impact the performance of the localized QoS routing scheme. There are a number of reasons why it is preferable to minimize the number of candidate paths used for routing. First, there is a considerable overhead associated with establishing, maintaining and changing of candidate paths. Second, the complexity of the scheme that distributes network traffic among multiple paths increases noticeably as the number of candidate paths increases. Third, there could be a limit on the number of explicitly routed paths such as label switched paths in MPLS [5] that can be setup between a pair of nodes. Therefore it is desirable to use as few paths as possible while at the same time minimizing the congestion in the network.

Under localized QoS routing schemes, the candidate path set remains static while their properties are adjusted dynamically according to the network dynamics. A network node in localized QoS routing schemes can predict and judge the quality of paths only by routing some traffic along them. Therefore, it is not possible to update the candidate path set based on local network state information. Beside that, due to changing network conditions, a small number of good candidate paths cannot be selected statically whereas dynamic selection of candidate paths requires global network state information updates. However, these updates would not cause important load on the network as long as their occurrence is not more than what is needed to convey connectivity information in traditional routing protocols like OSPF.

To accomplish the best performance of localized QoS routing schemes, there are a number of factors that must be considered in the candidate path selection process. The first factor is that a candidate path selection method must give higher priority to shorter paths because using a longer path for a connection requires additional network resources. The second factor is that a path selection method should attempt to balance the load of the network and so enhance the network performance. The third factor is the number of links shared by the candidate paths for a given source-destination pair should be minimized to allow the routing algorithm perform better. Therefore, in designing path selection schemes, these factors should be taken in consideration together and so trade-offs must be made [50].

3.2.1 Breadth first search path selection (BFS)

This breadth first search path selection (BFS) aims to find shortest paths between any source node and destination node to form a candidate path set. This method employs a breadth first search algorithm to find all minimum hop count paths and some alternative paths which have a limited extra path length (e.g. 1 hop more than the candidate paths set minimum-hop), for each source-destination pair. This method does not take in account the global load-balancing and shared link factors [50].

3.2.2 Per-pair path selection (PP)

The Per-pair path selection (PP) tries to discover shortest paths between each source-destination pair at the same time as reduce the number of common links in the candidate paths set for a specified source destination pair. This approach chooses candidate paths for each source-destination pair as the following:

1. It starts by assigning the same weight to all links in the network.
2. Then it uses the Dijkstra shortest path algorithm to find the first candidate path.
3. It increases the weights on all the links along the first candidate the path.
4. Then it repeats the process (step 2) to find all required candidate paths

The purpose of increasing the weights for the links in the selected candidate paths is to circumvent using the links that are in the selected candidate paths. Therefore, the number of common links along the paths for a source destination

pair is reduced. However, this method does not consider the global load balancing issue [50].

3.2.3 Global Path selection (GP)

The basic idea of Global Path selection (GP) is to choose paths for all source destination pairs in a network such that the load is equally distributed to all links. This is done as follows:

1. It assigns the same weight to all links in the network.
2. Then, it considers each source destination pair in a round-robin fashion. For the first pair of nodes, it selects one shortest path using the Dijkstra shortest path algorithm and increases the weights on the links along the selected path.
3. Then, it repeats (step 2) for all source destination pairs.

As the weights of the links on the selected paths, it is possible that the links in the chosen paths will not be selected again. Because GP selects candidate paths for all the source destination pairs, it is promising that the load on used links will be balanced given that the less loaded links are more likely to be incorporated in a new candidate path. Therefore, the load on each link is liable to be smooth across all links [50].

3.2.4 Hybrid per-pair/global path selection (PPGP)

The Hybrid per-pair/global path selection (PPGP) improves the path selection process of the GP method. The PPGP attempts to use at least one shortest path in the candidate path for each source destination pair and take into consideration the global load balancing. The PPGP is carried out by the following:

1. It uses the PP to select one shortest path for each source destination pair.
2. Then it assigns the weights according to the network load and finds alternative paths using the GP.

The PPGP method is an improvement of the GP method. Consequently, it enhances the network load balancing [50].

3.2.5 PP with global tuning (PPGT)

PPGT attempts to accomplish global load balancing by carrying out the following steps:

1. It selects the set of candidate paths using PP.
2. Because PP does not support global load balancing, it is possible that the load on some link is higher than the load on other links. Therefore, the PPGT tries to balance the load by removing paths that use the loaded links.
3. Step 2 is repeated until no more paths can be removed and the number of paths between each source destination pair is below an intended value.

In this method, if x paths are found between a source destination pair in a network, then the PP method is used to find 2x paths between the source destination pair and then begins the removing process [50].

In all the above path selection methods two parameters are used, the maximum number of candidate paths between source destination pair nodes and the extra path length. The maximum number of paths parameter controls the number of candidate paths to be found. The extra path length parameter is incorporated to the heuristics so that the heuristics will only find paths that are within the additional path length of the minimum hop candidate path [50].

3.3 Localized QoS Routing Algorithms

In this section a number of localized QoS routing algorithms will be discussed. The main aim of these algorithms is to enhance the performance of the network. This aim is achieved by avoiding the need for exchanging communication and process overheads.

3.3.1 Dynamic Alternative Routing Algorithm (DAR)

Although localised QoS routing is a relatively new approach in the context of computer networks, the idea of routing using only local information has been used in telephone networks [51, 52]. All these schemes make use of local feedback information regarding flow acceptance and rejection to route future flows. The Dynamic Alternative Routing algorithm (DAR) [52] is an interesting

scheme which shares some similarity with schemes to be described and has been used in the British Telecom Trunk network [53]. In this scheme, when a call has arrived the source node tries a direct single link between the source and the destination. If the call cannot be routed along the direct link path then a preferred two-link path is used to route the call. If the call cannot be routed along the preferred two-link path, the call is blocked and another two-link path is selected randomly from the set of all two-link paths to be the new preferred two-link paths. Figure 3.1 which is taken from [51] shows the flow diagram of this scheme. Note that routing using the preferred path is subject to the trunk reservation condition which limits the “knock-on” effect that will be described in subsequent sections.

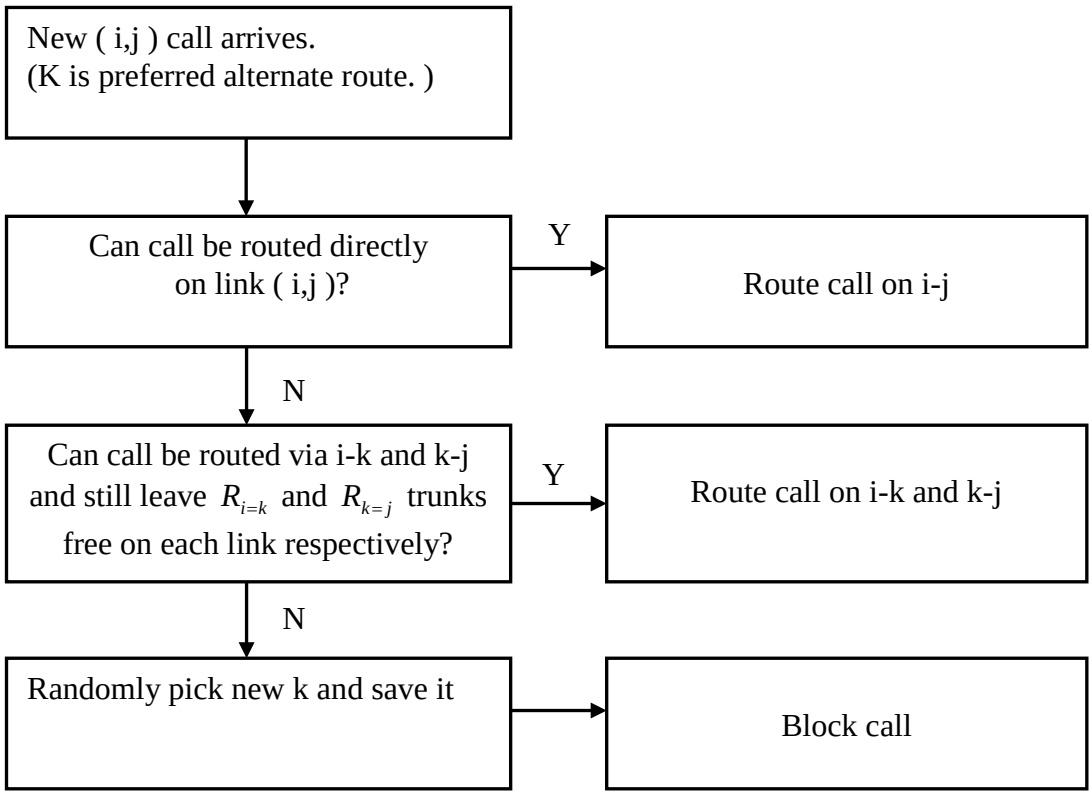
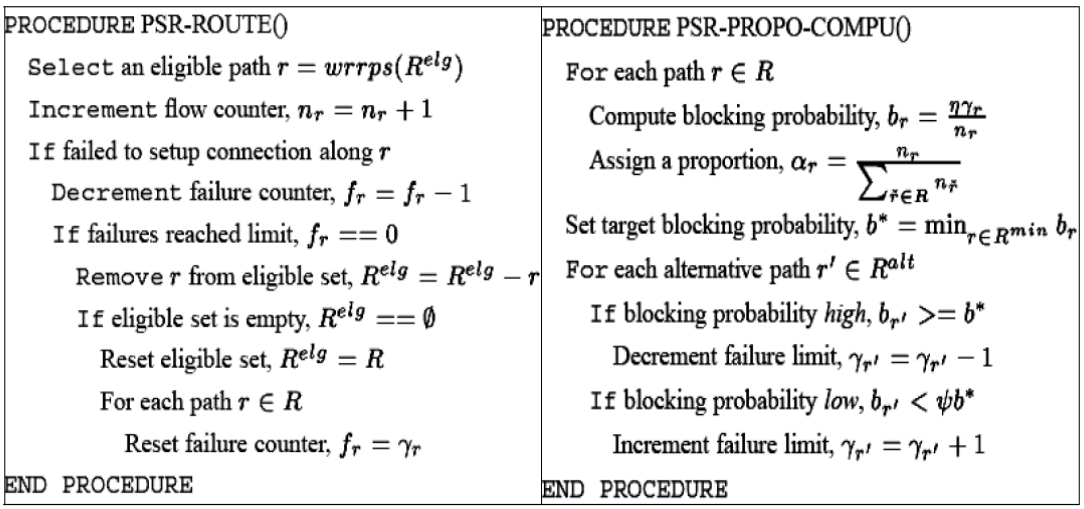


Figure 3.1 : Flow diagram for DAR algorithm [9]

3.3.2 Proportional Sticky Routing Algorithm (PSR)

The PSR scheme attempts to avoid using global network state information. The PSR algorithm is shown in figure 3.2. In this algorithm, for every pair σ, the source attempts to find the set of network traffic proportions $\{\alpha_1, \alpha_2, ..., \alpha_t\}$ such that the network traffic blocking rate of all paths in the candidate set *R* are stable ($\alpha_1 b_1 = \alpha_2 b_2 = ... = \alpha_{t-1} b_{t-1} = \alpha_t b_t$). The blocking probabilities (b_t) are collected locally and the proportions (α_t) are adjusted online to balance network traffic blocking rates. The justification behind this is to route more network traffic along a path with lower observed blocking probability and less network traffic to a path with higher blocking probability. PSR involves every

node to preserve a predetermined set of candidate paths *R* to each possible destination. Since it has been shown that routing algorithms that prefer short



(a) proportional routing (b) computation of proportions

Figure 3.2: Pseudo code for PSR [8]

paths usually outperform algorithms that do not take path length into consideration [28, 54], PSR differentiates between two types of paths, minhop paths (i.e shortest) R^{min} and alternative paths R^{alt}, where R = $R^{min} \cup R^{alt}$ which allows the algorithm to prefer minhop paths and limit the so called "knock-on" effect which is the cascade effect that results from using alternative paths which in turn forces other sources that use these alternative paths as their minhop paths to use their alternative paths [9, 55]. The PSR scheme (see Figure 3.2) can be viewed to operate in two stages: 1) proportional flow routing, and 2) computation of network traffic proportions. In the first stage, incoming network

traffic is routed along paths selected from a set of qualified paths R^{qua} . A path p is selected from the qualified paths set with occurrence determined by a prescribed proportion α_p. At the start, all the candidate paths are qualified and associated with a variable called the maximum allowed network traffic blocking parameter γ_p which determines the allowed amount of blocked network traffic routed along this path before it becomes disqualified. For each minhop path, γ_p is set to $\hat{y}$ which is a configurable system parameter. For each alternative path, the value of γ_r is dynamically adjusted between 1 and $\hat{y}$. A cycle is completed when R^{qua} becomes empty and a new cycle is started with R^{qua}= R**.** An observation period consists of η cycles; at the end of each observation period, a new network traffic proportion α_p for each path $p \in R$ is computed based on its observed blocking probability *bp*. Network traffic proportions for minhop paths are computed such that their flow blocking rates (*αpbp*) are equal. The minimum blocking probability among the minhop paths b^* is used as the reference to control network traffic proportions for the alternative paths. Which means, for each $p \in R^{alt}$, if $bp<\psi\ b^*$, γ_p =min(γ_p +1, $\hat{y}$). If $bp > b^*$, γ_p =max(γ_p - 1, 1),where ψ is a configurable parameter to limit the "knock-on" effect under system overloads. Note that $\gamma_p \geq 1$ ensures that some flows are routed along alternative paths to measure their quality.

3.3.3 Localized Credit Based QoS Routing (CBR)

The Credit Based Routing (CBR) [2] algorithm employs a straightforward routing procedure to route network traffic across the network. The CBR scheme uses a crediting scheme for each path in a candidate path set that rewards a path upon

network traffic acceptance and penalizes it upon flow rejection. The path choice relies on the path's credits: the path with the larger credits among the candidate paths is chosen to send the network traffic along. The CBR algorithm keeps updating each path's credits upon flow acceptance and rejection and does not compute a network traffic proportion. It also keeps monitoring the flow blocking probabilities for each path and adds this information to the crediting scheme to use in future path selection.

A set of candidate paths R between each source and destination nodes is required in the CBR algorithm. Like PSR, CBR predetermines a minhop path set R^{min} and an alternative paths set R^{alt} , where $R = R^{min} \cup R^{alt}$. CBR selects the largest credit path P.credits in each set, minhop paths set R^{min} and alternative paths set R^{alt} upon network traffic arrival. The network traffic is forwarded along the minhop path that has the largest credit P^{min} which is larger than the alternative path that has the largest credit P^{alt} ; otherwise the network traffic is forwarded along an alternative where $P^{min}.credits >= \Phi \times P^{alt}.credits$, where $\Phi \leq 1$. Φ is a system parameter that controls the usage of alternative paths. The CBR uses blocking probability in crediting schemes to enhance the algorithm performance, as a path with low blocking probability will gain more credits. Path credits are increased and decreased upon network traffic acceptance and rejection respectively using the blocking probability of the path.

However, CBR uses a MAX_CREDITS parameter to determine the maximum attainable credits for each path where $0 \leq$ credits $\leq$ MAX_CREDITS. The CBR algorithm records rejection and acceptance for each path and uses a sliding

window for a predetermined period of M connection requests. It uses 1 for flow acceptance and 0 for flow rejection, dividing the number of 0's by M to estimate each path's blocking probability for the period M.

Although CBR demonstrates better performance compared to PSR, the method applied for choosing paths to forward network traffic along in the algorithm, which relies on a crediting scheme, does not directly reflect the quality of that path. It would appear more logical to base the choice of path on the value of the required QoS metric like bandwidth in CBR; that is, directly on the path quality.

3.4 Summary

In this chapter we first gave a description of the localised quality of service routing approach in general, compared it to the global quality of service approach and outlined the differences between the two approaches. Next, we reviewed some existing techniques and related works in the context of telephone networks and computer networks.

Chapter 4:

Simulation Design and Implementation

4.1 Introduction

To study the behaviour of a communication system there should be a model that represents this system. There are two approaches to develop a model of a communication system: analytical modelling approach or simulation approach. The analytical modelling approach involves developing a solution for the concerned system. This solution is developed by using mathematical equations. These equations usually incorporate a number of numerical parameters which are called measures of performance of a system [56]. Sometimes it is quite difficult to develop such models due to the nature of the modelled system such as the type of routing algorithms. A simulation modelling approach involves developing a computer program to represent a communication system. Simulation is widely used in many areas including communication networks,

manufacturing, business, bioscience, military and health [57]. This reputation of simulation has resulted from the fact that a system can be studied under different conditions without the need to build a real system. Moreover, the wide availability of object oriented programming languages and support frameworks can be utilized to build simulation tools. Java is a powerful object oriented platform independent programming language that offers a lot of built in features[58].

In computer networks a network simulation is a technique where a computer program is developed to mimic the behaviour of a computer network. The need of computer networks simulation tools is obvious due to the growing complexity of computer network components. Beside that, computer network simulation provides researchers with powerful tools to study and understand these components and predict their behaviour.

4.2 Simulation of routing algorithms

Simulation of quality of service routing algorithms performance involves two tasks. The first task includes the simulation of the quality of service routing algorithm based on assumption on the computer network and parameter setting. The second task is about producing critical comments on the first task result, which should show the performance of the simulated quality of service routing algorithm [59]. The consistency and legitimacy of the second task is dependent on the first. An assumption and scenarios of the simulation

environment parameters close to those of a practical computer network, such as the Internet, is most preferred. In some situations, this is difficult since the Internet is complex and dynamic. Simulating a QoS routing algorithm in real computer networks such as the Internet environment requires complex analytical models of queuing delay at the packet level in the form of arrival and service processes [60], existence of QoS and best-effort network traffic with various connection duration profiles [61], synchronous runs of multiple scheduling policies [62] and congestion control mechanisms [63, 64] and also implementation of the proposed QoS architectures [8, 65-68]. Because quality of service routing is a network layer entity, a quality of service routing algorithm is unaware of many of these dynamic micro level conditions. The input to the quality of service routing algorithm would simply be the network state information either global or local and the graph structure that represents the underlying network.

In this chapter, the parameters and conditions adopted for simulating the proposed algorithms in Chapters 5, 6 and 7 will be discussed. The focus will be mainly on aspects of network graph models, settings and performance measures.

4.3 Graph Concepts

A graph is can be representation of a network and its connectivity A graph G(V, E) consists of two sets , a set of vertices (nodes) V and set of edges

(links) E. A graph is denoted by interconnected dots in a plane. A node v is a terminal point of a graph to represent a network component such as a router or switch. An edge is the link between two vertices.

Graphs can be classified into the following types according to the connection patterns among their nodes:

- Simple Graph that does not have loops or multiple edges, such as the graph in figure 4.1

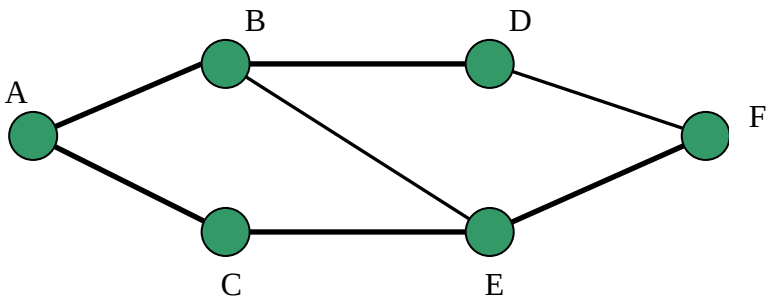


Figure 4.1: Simple graph

- Complete Graph is a simple graph in which each pair of distinct vertices are connected together as shown in figure 4.2. A complete graph has n(n-1)/2 edges, where n is the number of nodes.

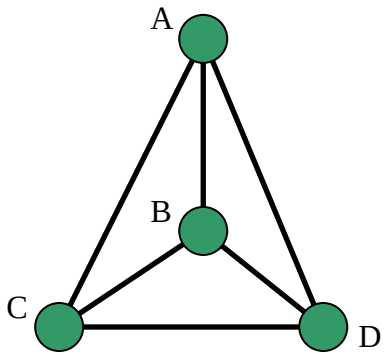


Figure 4.2 : Simple graph

- Connected Graph is a graph in which there is exist a path between each pair of nodes as shown in figure 4.1 and figure 4.2

- Regular Graph is a graph where each vertex has the same degree as shown in figure 4.3.

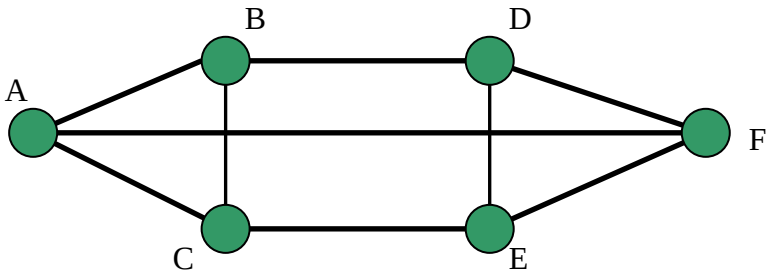


Figure 4.3 : Requlat graph

The graph efficiency and effectiveness can be determined by using a number of measures. These measures have been developed to measure the reality of a

graph structure. Furthermore, to compare and evaluate different graph structures. These measures include:

- Vertex degree is the number of edges connected to a vertex and it is simple, but effective measure of vertex importance. The higher its value, the more a vertex is important in a graph as many edges converge to it. A hub vertex has a high value, while terminal points have a value that can be as low as 1.

- Path length is the number of edges that composed a path. A path is a sequence of consecutive edges in a graph. A path connects two vertices in a graph.

- Graph diameter is the shortest path between the two widest separated vertices in a graph. This measures the topological length between two vertices. The diameter facilitates the measurement of the development of a network. The larger the diameter, the less linked a network tends to be.

- Graph order is the number of its vertices.

The representation of graph structure in simulation tools has an enormous effect on the performance and run time of a simulation tool. The following are some graph representations:

- Adjacency Matrix is a two dimensional array in which the elements indicate whether an edge is present between two vertices. If a graph has

N vertices, the adjacency matrix is an N x N array. Table 4.1 shows the adjacency matrix of the graph shown in figure 4.1

	A	B	C	D	E	F
A		1	1			
B	1			1	1	
C	1				1	
D		1			1	1
E		1	1			1
F				1	1	

Table 4.1: Adjacency matrix

- Incidence Matrix is a two dimensional array in which the elements indicate the relationship between a vertex and an edge is present. If a graph has N vertices and M edges, the incidence matrix is an N x M array, where rows represent vertices and columns represent edges. Table 4.2 shows the incidence matrix of the graph shown in figure 4.1.

	A	B	C	D	E	F
A	1	1	1			
B	1					
C		1		1		
D					1	
E			1		1	
F				1		1

Table 4.2: Incidence matrix

- Adjacency List is a representation of all edges in a graph. A graph representation can be done by using an array of length N (the number of vertices), every i-th element of the array being a list of the edges incident to vertex i. Table 4.3 shows the adjacency list of the graph shown in figure 4.1.

Vertex	Adjacent vertexes
A	B,C
B	D, E
C	E
D	B,F
E	B,C,F
F	D,E

Table 4.3: Adjacency List

- Incidence List is a graph representation that uses an array. Each element in the array corresponds to a single edge. Each edge contains a list of size two which corresponds to the end vertices of the edge.

4.4 Graph Models

A graph model is used in network simulations to represent the underlying network structure. It is important in simulating a quality of service routing algorithm to choose the graph model that is close to a real computer network topology. This is important because an algorithm performance would be acceptable in real computer networks such as the Internet may incorrectly indicate poor performance if the graph model that represents the network topology chosen for simulation is imprecise. For example, choosing a network topology with the size, node degrees and path lengths between pairs of nodes that are huge may turn into a polynomial time algorithm that is NP-complete [67]. A Graph model of a network topology indicates the pattern of link connectivity that connects nodes in the network, and the properties of links that form the interconnection. Because the computer networks are developing and evolving over time, it is very difficult to identify a distinctive and fixed graph model [69]. However, there are many graph models that can be used for simulation. The following sections list the main graph models widely used in simulations and comment on their suitability for the performance evaluations in Chapters 5,6 and 7.

4.4.1 Random Graph Models

Random graph models build a graph by adding links between pairs of nodes with a probability that is a function of the Euclidean distance of the nodes. They are the widely used models in performance evaluations of various quality of service routing algorithms and protocols. Random graphs are sometimes also called exponential graphs. There are at least five random graph models, each showing distinct topological properties [70]. The two most important random graph models are discussed which are the Waxman and the Doar-Leslie [71] graph models.

4.4.1.1 Waxman Graph Model

The Waxman Graph Model consists of a number of nodes (vertices) N. These nodes are distributed across a 2-dimensional Cartesian coordinate space. The placement of the nodes across the Cartesian space is uniformly random. Links (edges) are added by evaluating the probability functions (Eq. 4.1) of all possible pairs *(m, n)* of nodes, where *d(m, n)* is the Euclidean distance between node *m* and node *n*, *L* is the maximum possible distance between any pair of *(m, n)* nodes, β and α are controlling variables having the range $\beta \leq 1$ and $\alpha > 0$ [72].

$$p(m,n) = \beta e^{-d(m,n)/\alpha L} \tag{4.1}$$

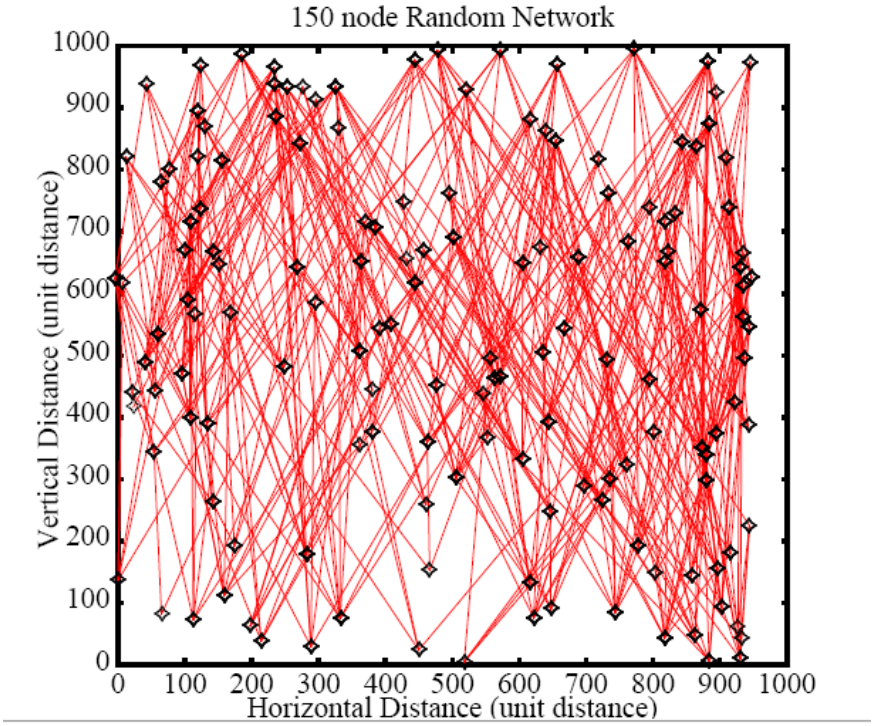


Figure 4.4 : a typical graph due to Waxman α= 0.25, β=0.3 (adapted from [72])

An increase in β increases the ratio of long links to short links. An increase in α increases the number of links in the graph. One problem with the Waxman model is there is no direct control over the proportional growth in N and node degrees. A graph generated with large N would have unrealistic node degrees.

4.4.1.2 Doar-Leslie Graph Model

The Doar-Leslie graph model [71] is an enhanced version of the Waxman model. The enhancement is due to modified probability function (4.2) that offers better control of granularity. A scaling factor $(kD)/N$ is added to ensure that mean degree of each node stays constant, where k is the scale factor and D is the mean degree of the nodes.

$$p(m,n) = \frac{kD}{N}\beta e^{-d(m,n)/\alpha L} \tag{4.2}$$

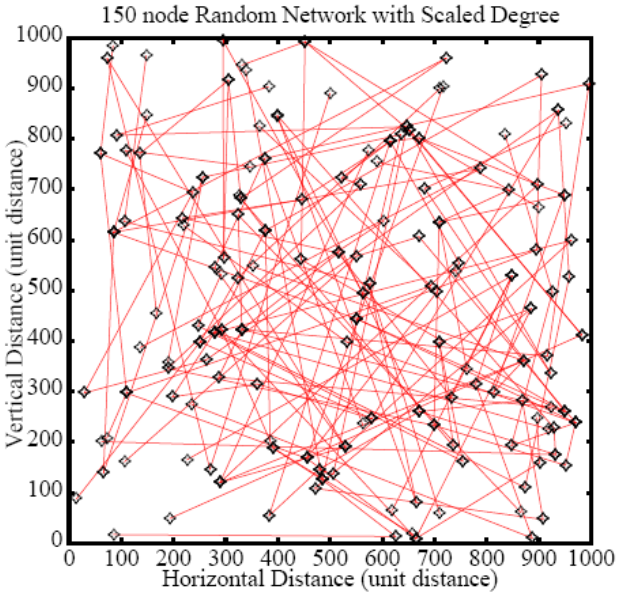


Figure 4.5 : Doar model (adapted from [72])

In Chapters 5, 6 and 7, the Doar-Leslie graph model will be used to generate random network topologies for simulation with certain topological characteristics. Nodes were placed randomly across a rectangular coordinate with a space dimension of 1000 × 1000 km^2. Each graph generated was tested to guarantee that each node is connected to at least one other node. The graph generated has reasonable distribution of short and long links, and with average node degree between 4 and 4.5. The values of k, β and α variables are taken from [59] as shown in table 4.4 to generate the required graphs.

N	K	β	α
20	27	0.15	0.15
40	27	0.13	0.16
60	27	0.12	0.17
80	27	0.12	0.18

Table 4.4: Variable values used to generate graphs (adapted from [59])

4.4.2 ISP Graph Model

Most Internet Service Providers (ISP) keep their network topological structure confidential for security purposes. These distinct ISP topologies are designed to optimize the network traffic performance, to ensure reliability and quality of each ISP Internet service offering and to guarantee security. An ISP topology can also be considered as a single autonomous system (AS) domain. There can be hundreds of interconnected routers and points of presence (POPs) [73, 74] within each ISP domain. The most widely used ISP topology for performance evaluation of quality of service routing algorithms [75, 76] is the modified ANSNET [77] as shown in Fig. 4.6.

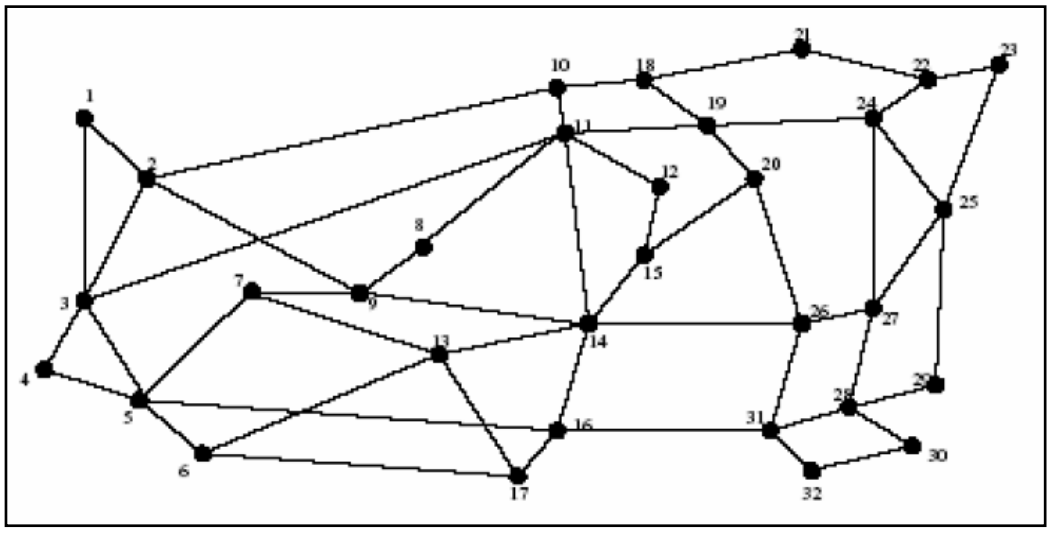


Figure 4.6 : Modified ANSNET ISP topology

ISP topologies were considered in the simulation simply because these graph models are widely used in research to evaluate the performance of the quality of service algorithms [10, 47, 78-80]. These graph models will be used in Chapters 5, 6 and 7 to evaluate the performance of the proposed localized quality of service algorithms.

4.4.3 Hierarchical Graph Model

Hierarchical graph models are fundamentally logical arrangements of the computer network into hierarchical structures. A famous example is the ATM PNNI structure [81]. Hierarchical graphs most fundamentally allow representation of the computer network as a collection of autonomous domains (AS). There are two methods of generating a hierarchical graph [82], the N-Level and the Transit-Stub method. Hierarchical graphs are considered in the performance evaluations of the proposed localized quality of service algorithm in Chapter 7.

4.4.3.1 Transit-sub model

The Transit-sub model consists of three levels corresponding to transit domain, stub domain and local area network domain. A transit domain consists of a set of backbone nodes. The purpose of the transit domain is to interconnect stub domains. While a stub domain is connecting local area networks (LANs) and control the traffic between LANs [83]. The transit domains generally correspond to wide area networks or metropolitan networks as shown in figure 4.7.

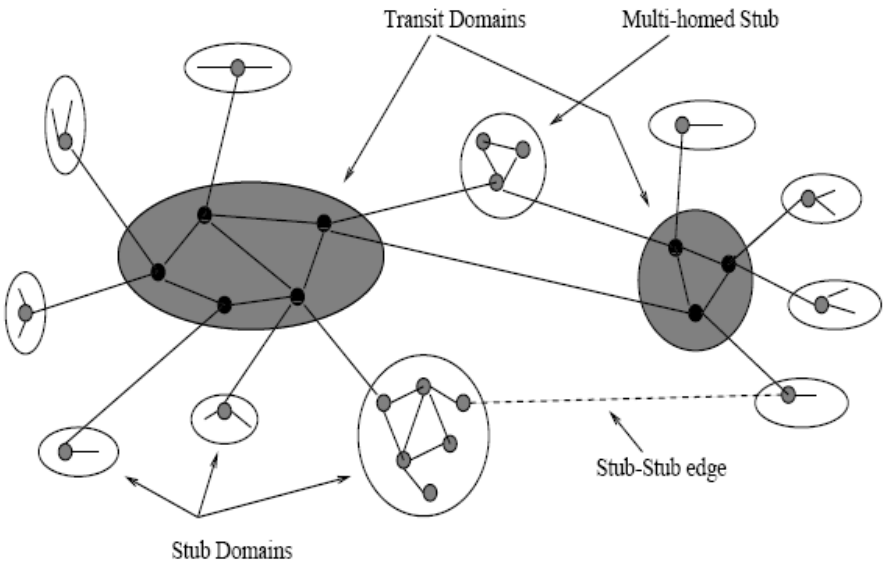


Figure 4.7 : Transit-stub structure (adapted from [83])

4.5 Simulator Design

The main aim in designing the simulator was to evaluate the performance of the localized quality of service schemes to be proposed using networks with a

moderate number of nodes (32-80). In addition, we would like to simulate the arrivals of very large numbers of flows (over 2M) in order to give accurate statistical results and to realistically simulate modern networks which have very large capacities and can accommodate very large numbers of network traffics.

In spite of the availability of simulation packages such as OPNET and NS-2, a Java based simulator has been developed to simulate the proposed algorithms.

The main reason for using Java instead of using the existing simulation packages is that these ready made packages require a considerable customization and the addition of missing components. Furthermore, due to their focus on low-level modelling (such as packet-level details), these packages suffers from scalability problems when studying large networks with large numbers of concurrent flows. Therefore, we preferred to develop our own simulator that provides the basic functionalities of modelling the network elements and enables us to simulate and study the proposed algorithms.

4.6 Simulator Structure

The functional diagram of the simulator is shown in figure 4.8. The diagram shows the key modules and gives an overview of the simulation process. The functions of each module are described below:

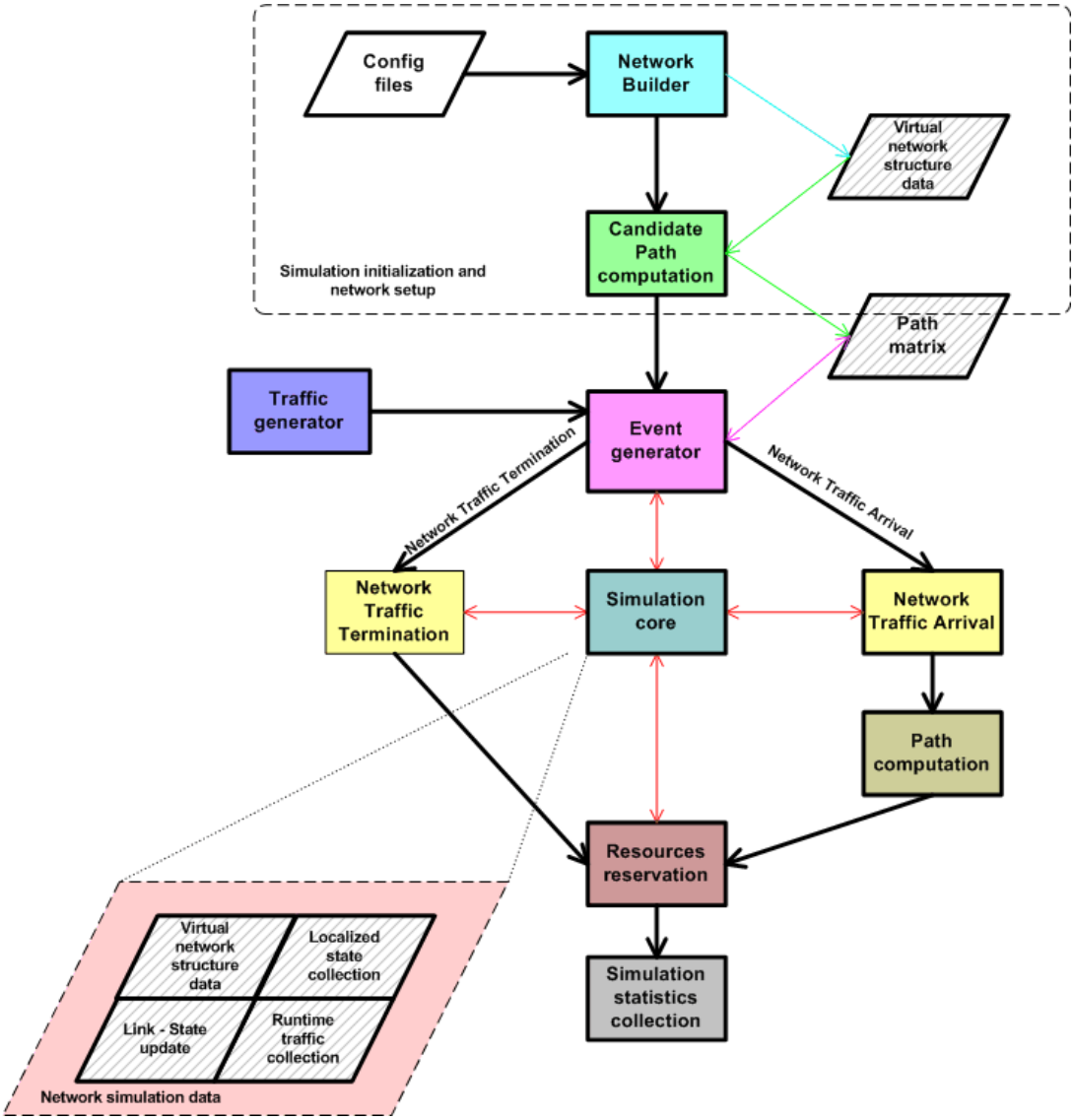


Figure 4.8 : Simulation functional diagram

Simulation initialization and network setup: This module carries out the initializing steps for the simulation. This includes a network builder module that reads network topology data, builds the network topology, sets link capacities and simulation parameters. Routing tables of all nodes are built during these steps. In addition, the candidate path set computation module generates candidate paths for each source/destination pair and is constructed based on virtual network structure data. This set is used by localised algorithms only.

Traffic Generator module: The traffic generator module generates the network traffic needed for network simulation. It determines the features of each arriving network traffic flow based on the specified traffic parameters. This module supports the following features:

- Arrival process: the network traffic can be modelled as a Poisson process.
- Flow duration: network traffic duration has an exponential distribution.
- Bandwidth requirement selection: is uniformly distributed.
- Source/destination selection: is chosen randomly.

Event Generation: The event generation module is responsible for generating the three main events of the simulator:

- Network traffic arrival: The arrival of a network traffic will activate the invocation of this handler which will pass this network traffic along with its bandwidth requirements *b* to the path computations module to determine the best apparently feasible paths. If the path computation module returns a path that is considered to be feasible, the handler will call up the network traffic signalling and resource reservation module to signal the network traffic.

- Network traffic termination: the function of this handler is to invoke the network signalling and resource reservation module to release any resources reserved for the terminated network traffic.

Simulation core: this module is responsible for:

- Localised state collection: This module is used by localised quality of service routing algorithms and is responsible for collecting local information required by these algorithms. Note that this module interacts with the flow signalling and resource reservation module to obtain the information regarding a flow's acceptance or rejection.

- Link-state update: This module is used by algorithms that require global state information, such as the WSP algorithm[84]. Normally, updating the global link network state information is done using flooding or spanning tree strategies implemented by a particular routing protocol. However, this will affect the scalability of the simulator due to the vast amount of detail and message passing that needs to be simulated in low-level detail. Since we are not focusing on the effect of these particular strategies, the link-state updater module does not implement any particular strategy. Instead, each link is associated with a variable that represents its advertised available bandwidth. This variable is updated periodically according to the update interval parameter to reflect the current available bandwidth. The path computation module will use this variable to perform path selection. Thus we are able to simulate the effect of out of date global state without affecting the scalability of the simulator.

- Virtual Network Structure Data: It is responsible for keeping all the network data and resources such as nodes, links that connect nodes, links capacities and paths.

- Runtime traffic Collection: This module is responsible for collecting runtime simulation statistics.

- Resources Reservation: this simulates the basic network traffic signalling and resource reservation mechanism. When the path computation module returns a path *p* that is considered to be feasible to support the arriving network traffic which has a requirement of *b* units of bandwidth, the network arrival handler invokes this module to signal that network traffic. The signalling mechanism is simulated at the packet-level where the source node initiates a hop-by-hop signalling to reserve bandwidth *b* on each link $\ell \varepsilon p$, each link ℓ in the network has available bandwidth $bw(\ell)$ which can be allocated to new network traffic. As the signalling message traverses the path *p*, each node performs an admission test to check that the link can truly support the network traffic. If the link has sufficient resources(i.e, $bw(\ell) \geq b$) the node reserves bandwidth *b* for the new network traffic (i.e., $bw(\ell) = bw(\ell) - b$) before passing the set-up message to the next link in the path *p*. The network traffic is accepted by the network if all the links in path *p* can support the network traffic , othrwise a failure message is propagated back to the source node releasing

previously reserved resources (i.e $bw(\ell) = bw(\ell) + b$) and the network traffic is rejected.

The module is also invoked by the network traffic termination handler when a flow is terminated to release any resources consumed by that flow (i.e., $bw(\ell)=bw(\ell)+b, \forall \ell \in p$).

Path computation module: This module implements four localised QoS routing algorithms, the Dynamic Path Substitution Localized QoS Routing Algorithm (DPSLRA), Disjoint Path Localized QoS Routing Algorithm (DPLRA), Hierarchical Localized QoS Routing Algorithm (HLRA), and Credit Based routing (CBR) algorithm , and one global QoS routing algorithm which is the WSP algorithm. The localized QoS routing algorithms use the information collected by the localised state collection module while WSP uses the global state supported by the link-state updater module.

Simulation statistics collection module: This module is responsible for collecting the simulation related statistics. This includes performance metric's statistics such as blocking probability, bandwidth blocking probability and average carried traffic. This module also produces these statistics as a function of time, so it is possible, for example, to plot the carried traffic in the time axis during the whole simulation or the utilization of a certain link in response to the rapid changes in the offered load.

4.7 Simulator Implementation

The simulator was built using Java programming language as shown in figure 4.9. The main reason for choosing Java is that it is an object oriented and platform independent programming language. It also contains a lot of packages that support network functions such as generating random variables that simulate network traffic.

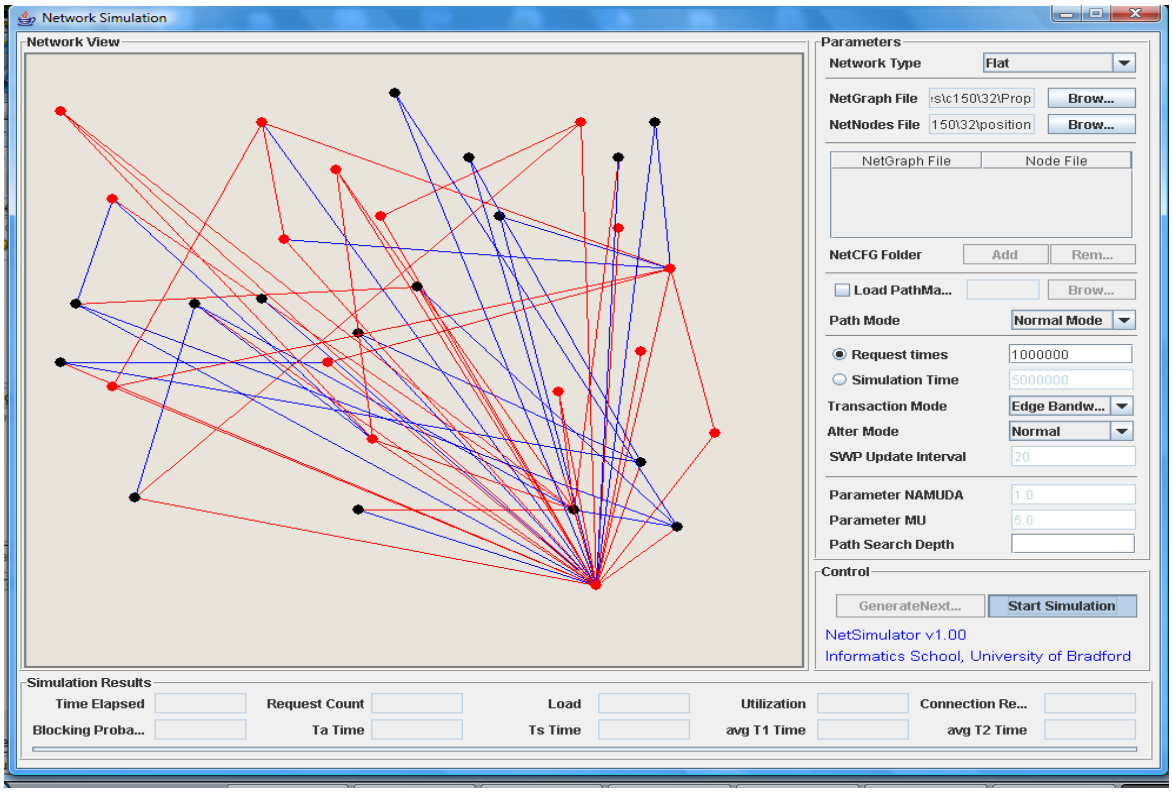


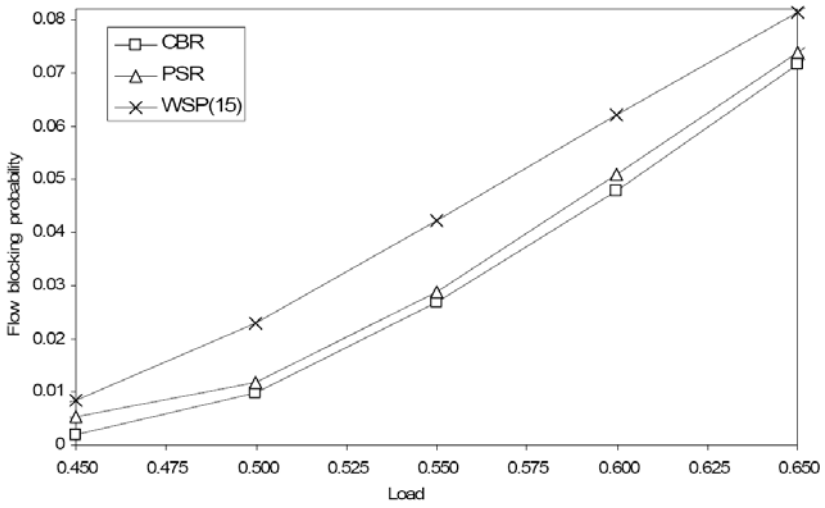
Figure 4. 9 : Network Simulator

4.8 Simulator validation

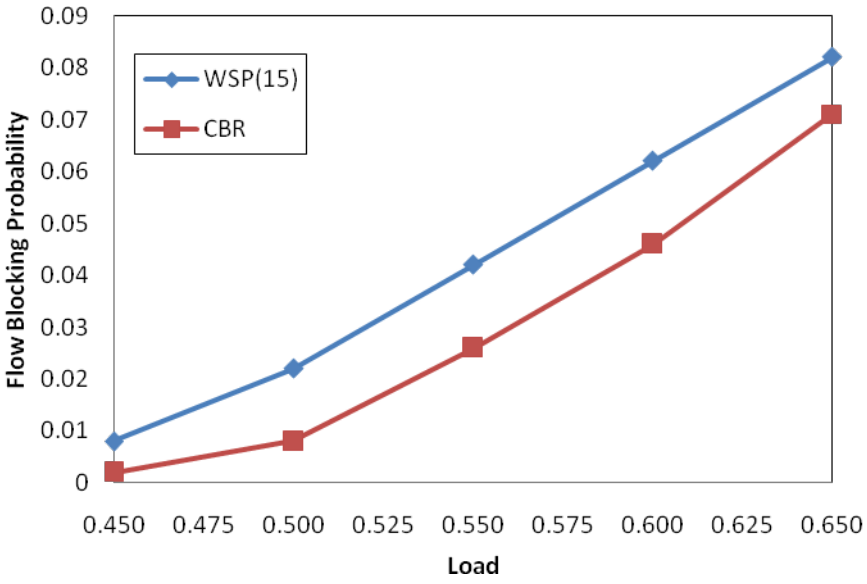
Simulation validation is the process of assuring that a model provides meaningful results for the situation being simulated[85]. To validate the algorithm's functionality, we implement the validation process as following:

1. For the CBR algorithm, using the same simulation configuration and parameters described in [86] by the developer of the CBR algorithm, we repeat the simulations using our simulator and found the result are very close to the results reported shown in Figure 4.10.

2. For the widest shortest path algorithm (WSP), the simulator validated by comparing the results of some simulation scenarios obtained by our algorithms with those obtained by another simulator called routesim [87].



(a) Original result taken from [86]



(b) Verified results

Figure 4.10 : Simulator validation

4.9 Summary

To study communication behaviour, there are two approaches: analytical modelling approach and simulation approach. A simulation approach is commonly preferred since that is capable of evaluating complex systems which are intractable by analytical approaches without simplifying assumptions that are often not realistic.

Simulation of quality of service routing algorithms performance involves two tasks. The first task includes simulation of the quality of service routing algorithm based on specific assumptions and parameter settings.

In order to carry out the simulation of routing algorithms, a network model should be used. There are a number of graph models that have been used to model a computer network. These graph models can be classifieds into types: flat models such as Waxman model and hierarchal model such as Transit-stub model.

Based on this background information a network simulator has been developed using Java programming language to simulate and study the performance of the proposed algorithms.

Chapter 5:

Dynamic Path Substitution Localized QoS Routing Algorithm (DPSLRA)

5.1 Introduction

The scalability problems related with the global network state information in conventional QoS routing approaches has motivated researchers to look for alternative approaches that can cope with the complexity and the dynamics of large scale networks.

In this chapter we introduce a new Dynamic Path Substitution Localized QoS Routing Algorithm (DPSLRA). This algorithm shows better performance when

compared to Credit Based Routing (CBR) and widest shortest path algorithm (WSP). First a detailed description of the proposed algorithm is presented then its performance is compared against Credit Based Routing (CBR) and widest shortest path algorithm (WSP) using bandwidth as a metric. After that the performance of the proposed algorithm will be compared to the CBR algorithm and widest shortest path algorithm (WSP) using the delay metric. Finally, a chapter summary will be presented.

5.2 Dynamic Path Substitution Localized QoS Routing Algorithm (DPSLRA)

5.3 Assumption

The following assumptions are made when designing the DPSLRA algorithm in bandwidth mode and delay mode.

Bandwidth Guarantee: Different applications may require different QoS requirements depending on their natures and needs. However, DPSLRA in bandwidth mode and delay mode assumes that these requirements can be translated in terms of bandwidth or delay, respectively. In the case of bandwidth, this could be the effective bandwidth, peak bandwidth or the average bandwidth that specifies the characteristics of the application's generated traffic.

In the case of delay, this is end-to-end packet delay on a path and this includes queuing delay, propagation delay and transmission delay.

In summary, the DPSLRA algorithm in bandwidth mode and delay mode will admit network traffic only if it finds a path that satisfies the network traffic requirements.

Network traffic flow signalling and resource reservation: The DPSLRA algorithm in bandwidth mode and delay mode is a route selection algorithm; therefore it assumes that the network is supported by a signalling and resource reservation mechanism to setup the actual path for the new network traffic flow. The signalling process starts at the source node by sending a setup message along the selected path. In the case of bandwidth, each intermediate node performs an admission test to see if the outgoing link has sufficient residual bandwidth for the new network traffic flow or not. If the link can accommodate the new network traffic flow, the requested bandwidth is reserved for that flow and the message is forwarded to the next link. The network traffic flow is admitted if all the links can support the network traffic flow, otherwise a failure message is propagated back to the source node and the network traffic flow is rejected. A similar process also applies to delay. In both cases, the information regarding network traffic flow acceptance or rejection must be accessible to the DPSLRA algorithm in order to collect flow statistics.

5.4 The DPSLRA (Bandwidth mode)

The proposed DPSLRA algorithm is like the Credit Based Routing (CBR) algorithm. It uses simple routing rules to compute routing paths across the network. It uses a very simple crediting scheme that rewards a path upon flow acceptance and penalizes it upon flow rejection.

DPSLR pseudo code is shown in figure 5.1. It is based on a simple idea that improves the performance of CBR. The proposed algorithm uses a simple mechanism based on path substitution. DPSLR, like CBR, requires every node to maintain a predetermined set of candidate paths **R** which contains minhop paths $R^{\min}$ and alternate paths R^{alt}. In addition to that DPSLR requires every node to maintain a set of reserved paths R^{res}. Hence, the set of candidate paths R = $R^{\min} \cup R^{alt} \cup R^{res}$. Every path P is associated with a variable P.credits that stores the accumulated credits gained so far.

The DPSLR keeps monitoring the credits of $R^{\min}$ (line 3) R^{alt}, (line 4) and R^{res} (line 5). The network traffic is routed along $R^{\min}$ if $R^{\min}$.credits >= Φ× R^{alt}.credits (lines 6-7), where Φ≤1, otherwise, R^{alt} is chosen (line 9). Φ is a system parameter that controls the usage of alternative paths and limits the "knockon" effect described earlier. According to [54], the value of Φ that gives good results is 1 or between .85 and .95 if the average number of minhop paths $R^{\min}$ is small compared to the average number of alternative paths R^{alt}. Note that, if there is more than one path with maximum credits, the first one is chosen.

```
Initialize

set   P.credits = Max_Credit, ∀   P ∈ R
DPSLRA(Max_Credit, Min_Credit)
1     if  P.credits =0     ∀P ∈ R
2           set   P.credits = Max_Credit, ∀   P ∈ R
3     P^min = max(P.credits : P ∈ R^min)
4     P^alt = max(P.credits : P ∈ R^alt)
5     P^res = max(P.credits : P ∈ R^res)
6     if  P^min.credits >= ΦP^alt.credits)
7           set  P = P^min
8     else
9           set  P = P^alt
10     route flow along path P.
11     if flow accepted
12          UpdateBlockingProbability(P,1)
13          amount = 1 - P.getBlockingProbability()
14          P.credits = min(P.credits + amount, Max_Credit)
15     else
16          UpdateBlockingProbability(P,0)
17          amount = P.getBlockingProbability()
18          P.credits = max(P.credits - amount,0)
19     if (average P.credit < Min_Credit)
20          set P = P^res.top
```

Figure 5.1 : pseudo-code for DPSLRA.

When the network traffic is accepted along the selected path, its blocking probability is updated accordingly (line 12) and P.credits is incremented by an amount that corresponds to its success probability (line 14). On the other hand, if the network flow is rejected its blocking probability is updated accordingly (line

16) and P.credits is decremented by an amount that corresponds to its blocking probability (line 18). The rationale behind using blocking probabilities in the crediting scheme is to improve the performance since paths with low blocking probabilities will receive more credits, hence, more network traffic will be routed along them and vice versa. In this way, the amount of network traffic routed along a certain path will be inversely proportional to its blocking probability. Also, the DPSLR algorithm keeps a history of credits of every $P \in R$ and uses this history to calculate the average of credits. This average will be used to govern the substitution of the path $P \in R^{min} \cup R^{alt}$ with a path $P \in R^{res}$ when this average is less than Min_Credit. For example, if a path $P_i \in R^{min} \cup R^{alt}$ has credits $(c_1, c_2, c_3, ...c_n)$ then the average of these credits is as shown in equation 5.1.

$$\text{Credit\_Average} = \frac{\sum_{i=1}^{n} c_i}{n} \qquad (5.1)$$

Max_Credit is a system parameter which determines the maximum attainable credits for each path, i.e. $0 \leq P.credits \leq Max\_credit$. This has the following advantages:

First, it stops a path with very low blocking probability from accumulating very large credits which could prevent the algorithm from selecting other good paths and causes undesirable performance. Second, if the quality of a path with very large credits degrades, it will take a long time to bring its credits down to the

correct level, hence, imposing this upper bound helps DPSLR to react and adapt in a timely manner. Third, it is more sensible to treat paths with credits beyond a certain level as if they have the same quality, even if some paths can attain more credits, this also ensures that the load will be balanced among these paths as their credits compete to reach this upper limit. Since DPSLR continuously monitors flow blocking probabilities, it records the information regarding the acceptance or rejection of flows for every path and uses a simple moving average with predetermined period to calculate the path's blocking probability. For a period of M, blocking probability of every path will be calculated using the most recent M flow data. This can be implemented easily by using a simple list with fixed size M. A new element is added to the beginning of the list after removing the oldest element from the list.

Min_Credit is a system parameter which is compared with the Average_Credit to control the path substitution process. Average_Credit value is less than Max_credit value and greater than Min_credit value (Min_Credit ≤ Average_Credit ≤ Max_credit)

```
UpdateBlockingProbability(P,x)
        If size_of(P.DataList) <  M
                Add x to the front of P.DataList
        Else
                Remove the oldest element from P.DataList
                Add x to the front of P.DataList
```

Figure 5.2 : Path blocking probability updating pseudo-code.

Figure 5.2 is shows the pseudo-code for updating the blocking probability of a path. As an example, let s={0,0,1,1,1} represent the information concerning the acceptance or rejection of the last M=5 connection requests, where 0 designates rejection and 1 designates acceptance. The oldest element is in the leftmost position where the newest one is in the rightmost position. The blocking probability will be 2/5. Now, if another connection request is accepted, then the oldest element will be deleted from s and 1 will be added to the rightmost position, i.e., s={0,1,1,1,1} and the blocking probability will be updated accordingly.

5.5 DPSLRA Performance Evaluation

In this section, the simulation setup will be presented and the performance of DPSLRA will be compared with the CBR scheme and widest shortest path algorithm (WSP).

5.5.1 Simulation setup (Bandwdith)

In the simulation of DPSLRA bandwidth mode, links between network nodes are assumed to be bidirectional and have the same capacity C in each direction (C=150Mbps). The network topology remains fixed during each experiment. Therefore, link failures are not considered. The network traffic arrives to each source node according to a Poisson process with rate λ and destination nodes are

selected randomly. Each node can act as source and/or destination. Applying this uniform random selection of destinations results in a uniform traffic in regular topologies, and non-uniform traffic in random and ISP topologies, this allows us to evaluate the performance under balanced and unbalanced loads.

Although Poisson traffic is widely used to model network arrivals, studies [88-90] showed that the network traffic arrival process is bursty and that distributions with heavy tails, such as Weibull, yield better models. Consequently, we also study the effect of bursty traffic where flow inter-arrival times follow a Weibull distribution.

Network traffic duration is exponentially distributed with mean 1/μ, while network traffic bandwidths are uniformly distributed within the interval [0.1-2MB]. To compute the offered network load references [30, 54] use equation (5.2).

$$\rho = \frac{\lambda N \overline{b} \overline{h}}{\mu L C} \qquad (5.2)$$

N is the number of nodes, $\overline{b}$ is the average bandwidth required by network traffic, $\overline{h}$ is the average path length (in number of hops) and L is the number of links in the network. Since the performance of routing algorithms may vary across different load conditions, our simulation experiments consider a wide range of loads to assist in evaluating the algorithms under different load conditions. However, under low loads, flows are almost always accepted, which results in very low blocking probabilities and all algorithms behave the same. Therefore, we try to chose the most relevant range of loads and omit the rest

either because the relative performance of the algorithms is reflected in the chosen range or the difference in the performance is insignificant.

The parameters for algorithms are Max_Credit=5 and Φ=1 unless otherwise stated. Blocking probabilities are calculated based on the most recent 20 network connection requests.

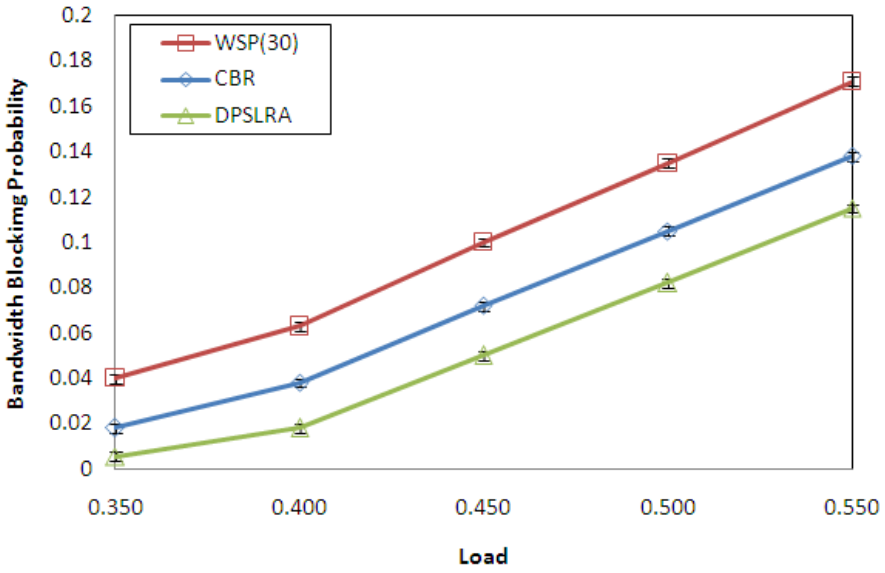
The set of candidate paths is chosen as shown in [9] such that, for each source destination pair, all the paths between them whose length is at most one hop more than the minimum number of hops are included in the set. Each run simulates the arrival of 2,000,000 network flows and the simulation results are collected after the first 500,000 connection requests.

5.5.1.1 Network traffic blocking probabilities

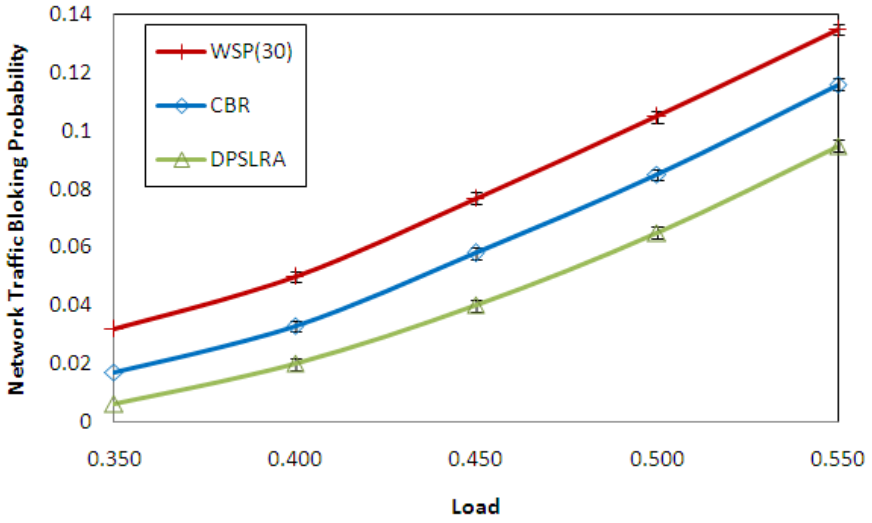
The first simulation experiment described is to compare the performance of the CBR, the DPSLRA and the WSP in terms of network traffic and bandwidth blocking probabilities under different load conditions. Figure 5.3 shows the performance of the three algorithms in terms of network traffic and bandwidth blocking probabilities under different load conditions. This experiment is repeated 10 times and the 95% confidence intervals calculated. The confidence intervals are very small, as shown in figure 5.3.

The overall network traffic and bandwidth blocking probabilities are plotted against the offered load using the random topology RAND80. We note that:

First, under low loads (load $\leq$ 0. 35), the difference in the performance (in terms of both metrics) of the three algorithms is relatively small which is logically expected since the probability of finding sufficient available bandwidth in each link is high and network traffic is almost always accepted. However, performance varies significantly when the load increases, and the bandwidth blocking probability grows more rapidly than the network traffic blocking probability, implying that network traffic with large bandwidth requirements are hard to route, as expected. Second, the relative performance of the three algorithms is the same for network traffic and bandwidth blocking probabilities suggesting that both metrics can be used to evaluate the performance of routing algorithms especially when the bandwidth requirements are small compared to link capacities. Third, the performance of the WSP algorithm is affected by the update period and it gives the worst performance and its blocking probability increases even under small load. Fourth, the DPSLRA scheme gives the best performance under all conditions.



(a) Bandwidth blocking probabilities



(b) Network traffic blocking probabilities

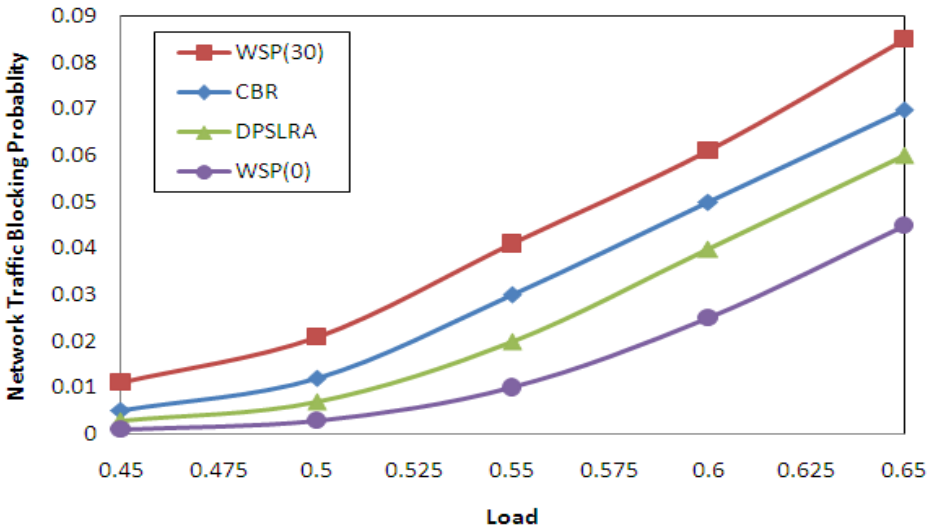
Figure 5.3 : Bandwidth and network traffic blocking probabilities for RAND80

There are many reasons for the above notes: for the WSP scheme paths are selected based on the QoS global network state information which is updated periodically and when periodic updates do not respond quickly to variations in network resources the performance will be affected. The CBR and the DPSLRA

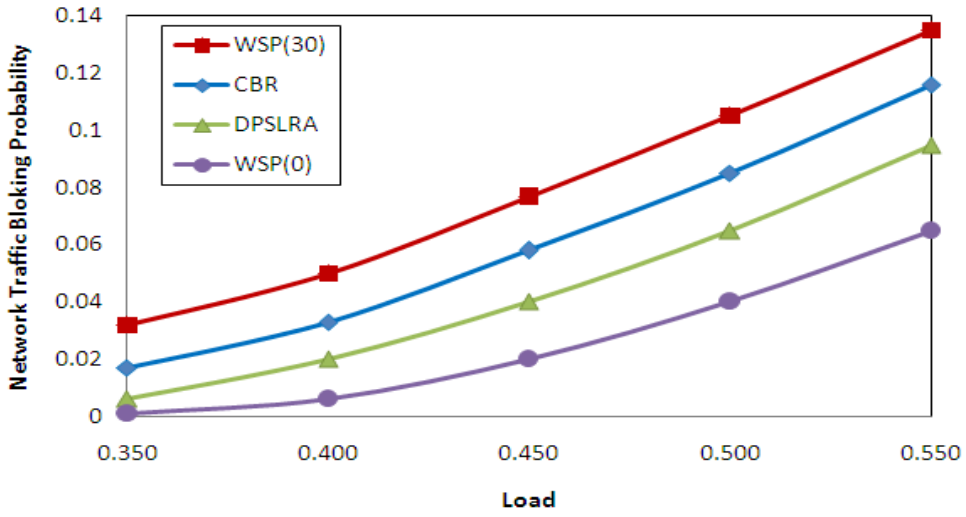
algorithms share almost the same mechanism to select the path with the maximum credits as long as it does not reject flows, however since credits of the selected path are updated after every network traffic routed along that path, any flow rejection will cause its credits to be decreased and an alternative path with more credits to be selected. The advantage of the DPSLRA that makes it superior to CBR is that it uses the substitution of the loaded candidate paths with unloaded paths form the reserved path set. The DPSLRA monitors the minimum candidate path set and the alternative candidate path set. If any of those paths get saturated then it will be substituted with the top path in the reserved path set.

5.5.1.2 Impact of network topology

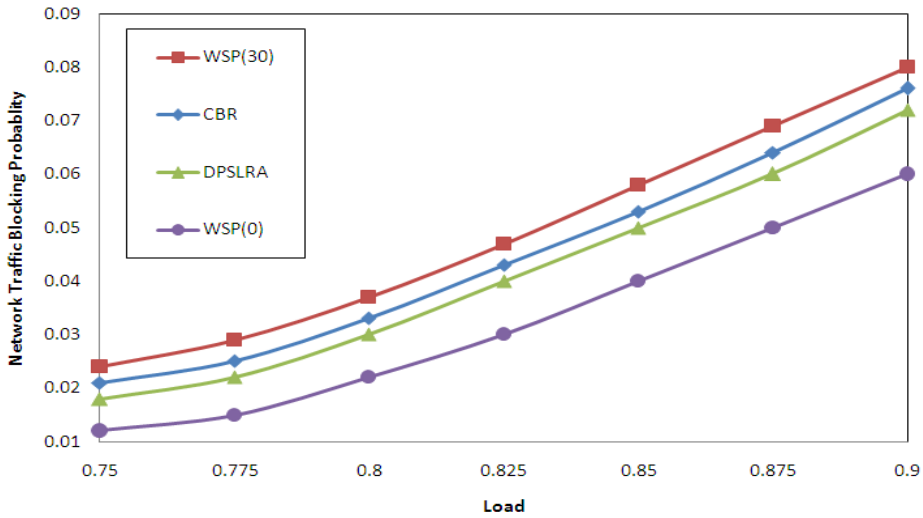
The network scale affects the performance of routing algorithms. Even across the same topology, different locations of sending and receiving nodes will have an impact on the performance of the routing algorithms [91]. The second simulation experiment is to evaluate the performance of the proposed algorithm using different types of network topologies to demonstrate that it can perform well across a wide range of network topologies.



(a) ISP topology



(b) Random topology (80 nodes)



(c) Random topology (32 Nodes)

Figure 5.4 : Impact of network topology

Figure 5.4 shows the network traffic blocking probability for the three algorithms using the network topologies described in the pervious chapter. Observing this figure we can see that the DPSLRA performance is better than the CBR algorithm and the WSP algorithm except when WSP has an unrealistically small update interval. The update interval of 0 in figure 5.4 for WSP indicates instantaneous update, which clearly gives an optimum bound on performance.

5.5.1.3 Impact of large bandwidth

To examine the performance the proposed algorithm under a range of bandwidth requirements we study the outcome of routing large bandwidth network traffic by considering the case of a mixed network traffic that a varying mix of small bandwidth network traffic and large bandwidth network traffic. The amount of bandwidth requests of network traffics are derived uniformly from a range [0.1-2MB] for small bandwidth network traffic with mean b1=1.05 for small-bandwidth network traffic, and [2-4MB] with mean b2=3 for large bandwidth network traffic, both types have the same flow duration.

Both types have a network traffic duration which is exponentially distributed with mean equal to 190 time units. The performance is measured by varying the fraction of small-bandwidth network traffic f while keeping the offered load fixed at $\rho = 0.8$ by changing the arrival rate according to equation (5.3).

$$\rho = \frac{\lambda(fb_1 + (1-f)b_2)N\bar{h}}{\mu LC} \qquad (5.3)$$

Figure 5.5 shows the network traffic blocking probability plotted as a function of the fraction of small bandwidth network traffic for RAND32.

Consequently, as the fraction of small bandwidth network traffic increases the blocking probability decreases and again, as expected, DPSLRA gives better performance than WSP for traffic composed mostly of small bandwidth flows.

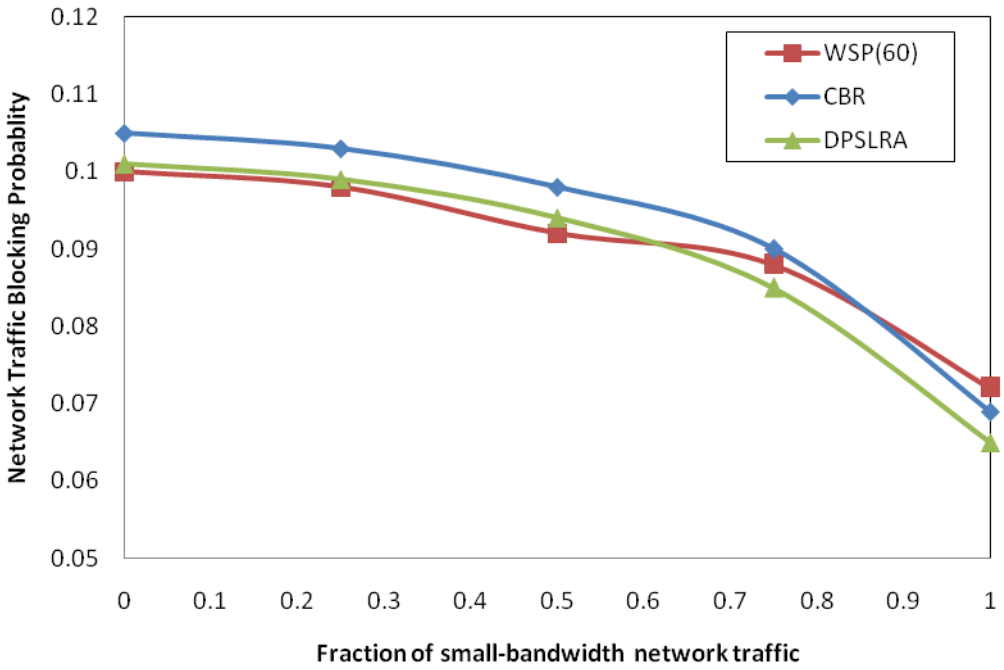
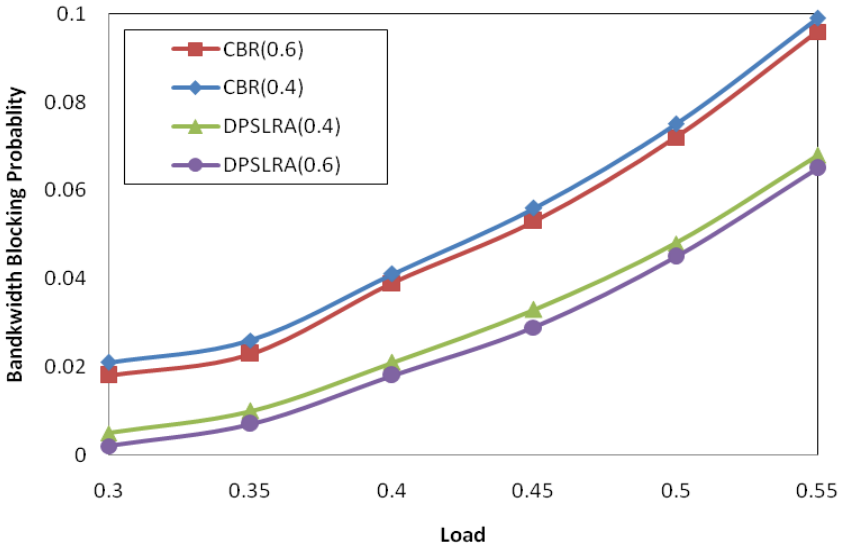


Figure 5.5 : Impact of large bandwidth connection requests.

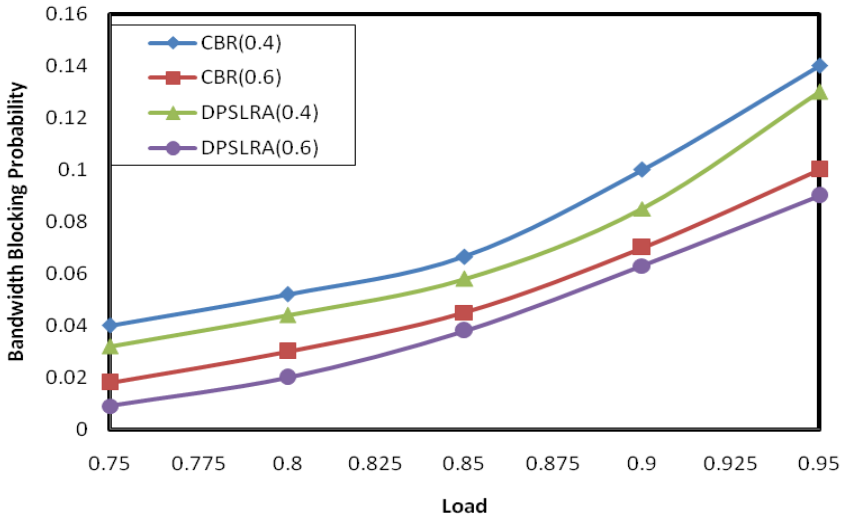
DPSLRA also performs well compared to CBR despite the fact that routing is independent of the amount of bandwidth requested. This remains true as long as the amount of bandwidth requested is small compared to the link capacities.

5.5.1.4 Impact of bursty traffic

Following [54, 88] we model bursty traffic using the Weibull distribution with two different values of the shape parameter of the distribution a=0.4 for large shape parameter and a=0.6 for small shape parameter where burstiness is increased with a smaller shape value. Figure 5.6 shows the network traffic blocking probability plotted against the offered load for two random topologies, RANDOM80 and RANDOM32, as we can see in the figure increased burstiness in the traffic arrival process results in increased blocking probability over the range of loads used.



(a) Random topology (80 Nodes)



(b) Random topology (32 Nodes)

Figure 5.6 : Impact of bursty traffic

5.5.2 The proposed algorithms (Delay mode)

In this section we modify the CBR and DPSLRA to use end-to-end delay as a quality of service metric. The algorithms guarantee that the end-to-end delay which the network traffic experiences from its source node to its destination node will not exceed a predefined constraint value, nor will any accepted flow

jeopardise the delay requirements of any existing path. In both modified versions, for each connection request the source node sends a connection request setup message along the selected path. The aim of this message is to check the availability of the required delay. Each intermediate node in the selected path performs a control test to find out the accumulated delay so far according to equation (5.4).

$$\sum_{j=1}^{m} \text{delay(j)} \text{ and existing delays} \leq \text{Delay\_constraint} \qquad (5.4)$$

Where m is the number of links in the selected path and j is the link index. Existing delays represent the tentative delay values of current connection requests if the new request is accepted. Delay constraint is the maximum required delay for a connection request. If the accumulated delay is less than the required QoS delay then the delay will be reserved and the message is forwarded to the next node until it reaches the destination node. This means that the connection request is accepted.

Otherwise, if the delay in any intermediate node is more than the required QoS delay a failure message will be send back to the source node and the intermediate nodes that have been traversed will release the reserved delay and the connection request will be rejected.

5.5.2.1 Credit Based Routing Algorithm Delay mode (CBRD)

The pseudo code for CBRD algorithm is as follows in figure 5.7: As we can see the CBRD is like CBR, it requires every node to maintain a predetermined set of candidate paths R to each possible destination. These paths are divided into two

sets: the set of minhop paths $R^{\min}$ and the set of alternative paths R^{alt} , where R = $R^{\min} \cup R^{alt}$. Each path P ∈ R has a variable called P.credits that stores credits for each candidate path. It is set to Max_Credit in the start of simulation. It is a system parameter that determines the maximum credit each candidate path can gain. When a connection request is initiated CBRD selects two paths from the candidate path sets, $P^{\min} \in R^{\min}$ (line 3) and $P^{alt} \in R^{alt}$ (line 4), which are the paths with maximum credits in each set.

```
Initialize
set  P.credits = Max_Credit, ∀  P ∈ R
DPSLRA(Max_Credit)
1     if  P.credits =0    ∀P ∈ R
2          set  P.credits = Max_Credit, ∀  P ∈ R
3     P^min = max(P.credits : P ∈ R^min)
4     P^alt = max(P.credits : P ∈ R^alt)
5     if  P^min.credits >= ΦP^alt.credits)
6          set  P = P^min
7     else
8          set  P = P^alt
9     route flow along path P.
10     if ∑Delay(s,d) and existing_delays ≤ Delay_constraint
11          UpdateBlockingProbability(P,1)
12          amount =1- P.getBlockingProbability()
13          P.credits = min(P.credits + amount, Max_Credit)
14     else
15          UpdateBlockingProbability(P,0)
16          amount = P.getBlockingProbability()
17          P.credits = max(P.credits - amount,0)
```

Figure 5.7 : The pseudo code for CBRD

The network traffic is routed along the path with the largest credit. If $P^{\min}$.credits >= $P^{\min}\Phi x P^{alt}$.credits (lines 5-6), where $0 < \Phi \leq 1$, then $P^{\min}$ is selected to route network traffic. Otherwise, P^{alt} is selected (line 8) to route network traffic. Φ is a system parameter that controls the usage of alternative paths and limits the 'knock-on' effect. The network traffic is accepted when the sum of end-to-end delay along the selected path and the delays of existing connection requests are less than or equal to the required QoS delay (line 10) which is controlled by equation (5.4). The blocking probability for the selected path is updated by increasing its credit with an amount that corresponds to its success probability (line 11- 13) when the network traffic is accepted. On the other hand, if the network traffic is rejected, the blocking probability for the selected path is updated by decreasing its credit with an amount that corresponds to its failure probability (line 15-17). The blocking probability update procedure is as described in section 5.4.

5.5.2.2 DPSLRA (Delay Mode)

The DPSLRA algorithm in the delay mode is like the Credit Based Routing (CBR) algorithm in delay mode. DPSLRA in delay mode pseudo code is shown in figure 5.8. It is based on a simple idea that improves the performance of CBRD. The proposed algorithm uses a simple mechanism based on path substitution.

DPSLR in delay mode is like CBRD, Every node has to maintain a predetermined set of candidate paths R which is divided in to three sets: minhop

set $R^{\min}$, alternate set R^{alt} and reserved set R^{res}. Therefore, the predetermined set of candidate paths R = $R^{\min} \bigcup R^{alt} \bigcup R^{res}$. Every path P is associated with a variable P.credits that stores the accumulated credits gained so far. The DPSLR in delay mode monitors the credits of $R^{\min}$ (line 3) R^{alt}, (line 4) and R^{res} (line 5). The network traffic is routed along $R^{\min}$ if $R^{\min}$.credits >= $\Phi \times$ R^{alt}.credits (lines 6-7), where $0 < \Phi \leq 1$, otherwise, R^{alt} is chosen (line 9). Φ is a system parameter that controls the usage of alternative paths and limits the "knockon" effect. Reference [54] stated that, the value of Φ that gives good results is 1 or between .85 and .95 if the average number of minhop paths $R^{\min}$ is small compared to the average number of alternative paths R^{alt}. Note that if there is more than one path with maximum credits, the first one is chosen.

The network traffic is accepted along the selected path that satisfies the required end-to-end QoS delay and does not jeopardize the end-to-end delay requirements of the existing connections. The blocking probability of the used path is updated accordingly (line 12) and P.credits is incremented by an amount that corresponds to its success probability (line 14). On the other hand, if the network flow is rejected its blocking probability is updated accordingly (line 16) and P.credits is decremented by an amount that corresponds to its blocking probability (line 18). The DPSLR algorithm in delay mode keeps a history of credits of every $P \in R$ and uses this history to calculate the average of credits. This average will be used to govern the substitution of the path $P \in R^{\min} \cup R^{alt}$ with a path $P \in R^{res}$ when this average is less than Min_Credit. For example, if a path $P_i \in R^{\min} \cup R^{alt}$ has credits $(C_1, C_2, C_3, ...C_n)$ then the average of these credits

is as shown in equation 5.1.

```
Initialize

set   P.credits = Max_Credit, ∀  P ∈ R
DPSLRA(Max_Credit, Min_Credit)
1     if  P.credits =0    ∀P ∈ R
2           set   P.credits = Max_Credit, ∀  P ∈ R
3     P^min = max(P.credits : P ∈ R^min)
4     P^alt = max(P.credits : P ∈ R^alt)
5     P^res = max(P.credits : P ∈ R^res)
6     if  P^min.credits >= ΦP^alt.credits)
7           set  P = P^min
8     else
9           set  P = P^alt
10     route flow along path P.
11     if ∑Delay(s,d) and existing_delays ≤ Delay_constraint
12          UpdateBlockingProbability(P,1)
13          amount = 1 - P.getBlockingProbability()
14          P.credits = min(P.credits + amount, Max_Credit)
15     else
16          UpdateBlockingProbability(P,0)
17          amount = P.getBlockingProbability()
18          P.credits = max(P.credits - amount,0)
19     if (average P.credit < Min_Credit)
20          set P = P^res
```

Figure 5.8 : The pseudo code for DPSLRA

Max_Credit is a system parameter which determines the maximum attainable credits for each path, i.e. 0 ≤ P.credits ≤ Max_credit . This has the following advantages:

First, it stops a path with very low blocking probability from accumulating very large credits which could prevent the algorithm from selecting other good paths and causes undesirable performance. Second, if the quality of a path with very large credits degrades, it will take a long time to bring its credits down to the correct level, hence, imposing this upper bound helps DPSLR delay mode to react and adapt in a timely manner. Third, it is more sensible to treat paths with credits beyond a certain level as if they have the same quality, even if some paths can attain more credits, this also ensures that the load will be balanced among these paths as their credits compete to reach this upper limit. Since DPSLR continuously monitors flow blocking probabilities, it records the information regarding the acceptance or rejection of flows for every path and uses a simple moving average with predetermined period to calculate the path's blocking probability. For a period of M, blocking probability of every path will be calculated using the most recent M flow data. This is can be implemented easily by using a simple list with fixed size M. New element is added to the beginning of the list after removing the oldest element from the list.

The system parameters Min_Credit, Average_Credit are used to control the path substitution process.

5.5.2.3 Performance Evaluation (Delay mode)

In this section, the simulation setup will be presented and the performance of DPLRA in delay mode will be compared with the CBRD scheme.

5.5.2.4 Simulation setup

In each simulation experiment the network topology is fixed during the simulation period. The network traffic form source node to destination node is generated with uniform probability. The network traffic's interarrival times at the source node are exponentially distributed with the mean $1/\lambda$. The mean of the network traffic holding time is$1/\mu$, with an exponential distribution. The required QoS delay is varied, ranging between 15 and 30 time units. It is assumed that all links in the topologies are bidirectional and the mean delay time for each link is also exponentially distributed.

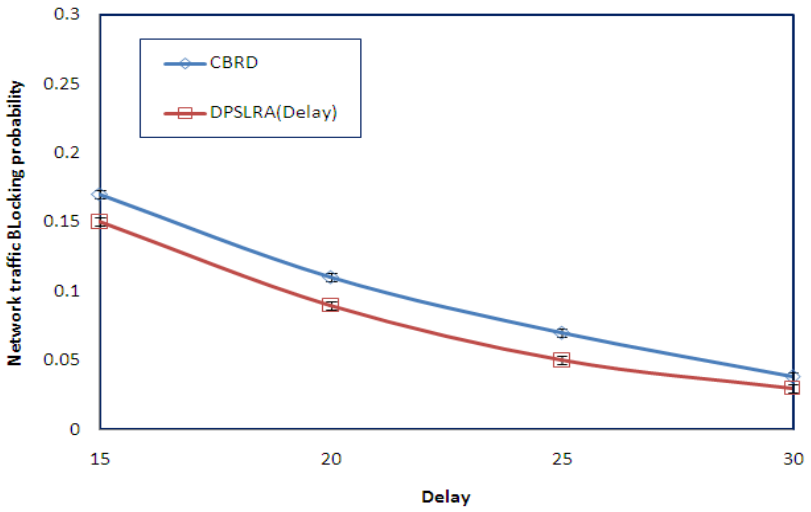
The system parameters used in the simulation are Max_Credit=5 and Φ=1. Blocking probabilities are calculated based on the most recent 20 flows for both algorithms. Candidate paths between each source-destination pair in a network topology are chosen, so we include minimum hop and minimum hop +1 in the set to get the required number of candidate paths between each pair. All experiments simulate 2,000,000 connection requests and results collected after a warm-up period of 500,000 connection requests.

Network traffic blocking probability was used to measure the performance of the algorithms. The blocking probability is defined as in equation 5.5:

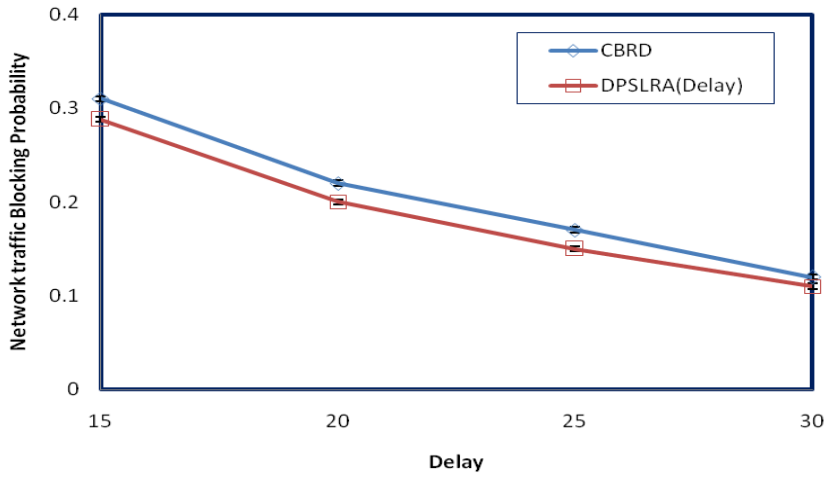
$$\text{blocking probability} = \frac{\text{No. of rejected connection requests}}{\text{Total no. of connection requests}} \qquad (5.5)$$

5.5.2.5 Impact of delay constraint for different topologies

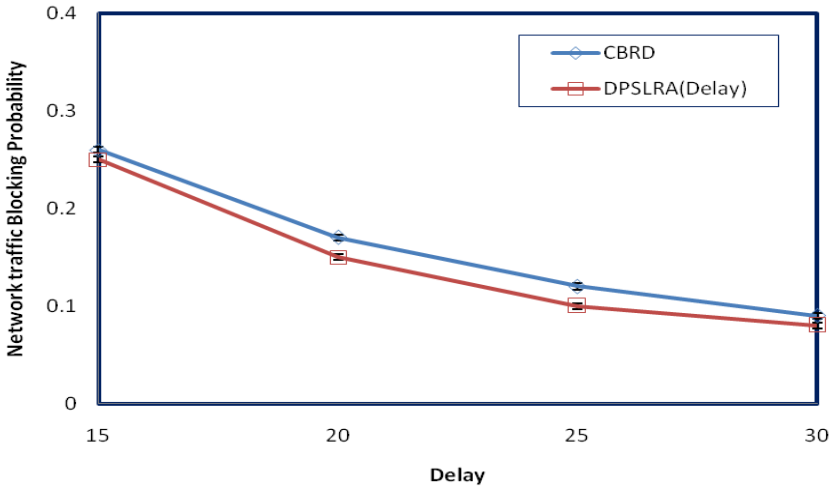
In this section the effect on the performance caused by the delay constraint will be shown for different topologies. Figure 5.9 illustrates the network traffic blocking probability plotted against various delay constraints for different types of network topologies. It can be noted that all algorithms satisfy most flows under large delay constraint (constraint 30), which can be expected as the probability of finding a path that satisfies a large delay constraint is high and most network traffic will be accepted. In figure 5.9 (a) both algorithms give good performance with the same constraint. This is due to the fact that ISP topology has the smallest average path length. However, performance under some constraints may be significantly degraded based on the average path length in the topology. This can be noticed in figure 5.9 (b), with Lattice topology giving the lowest performance compared with the other topologies under a delay constraint of 30. This is because Lattice topology has the largest average path length.



(a) ISP topology



(b) Lattice Topology



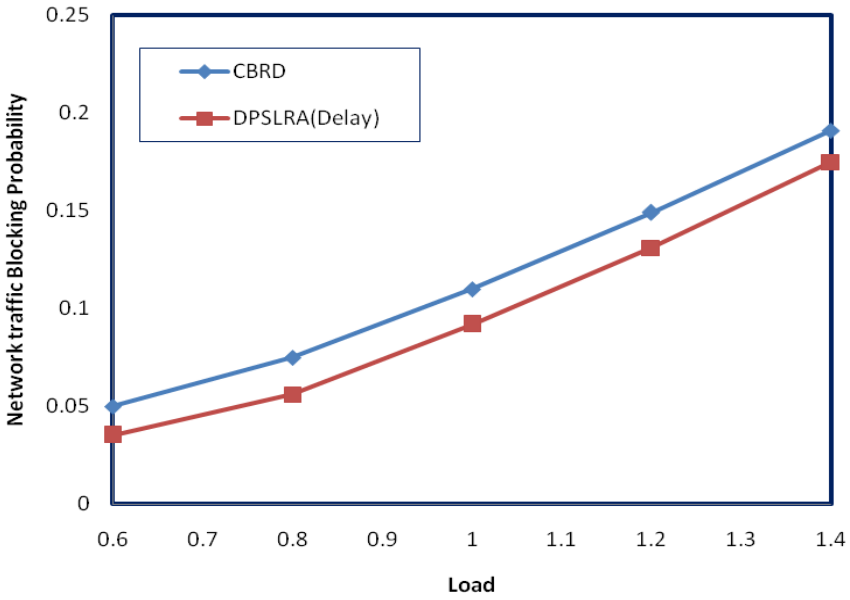
(c) Torus Topology

Figure 5.9 : Network traffic blocking probability under various network topologies and various delays

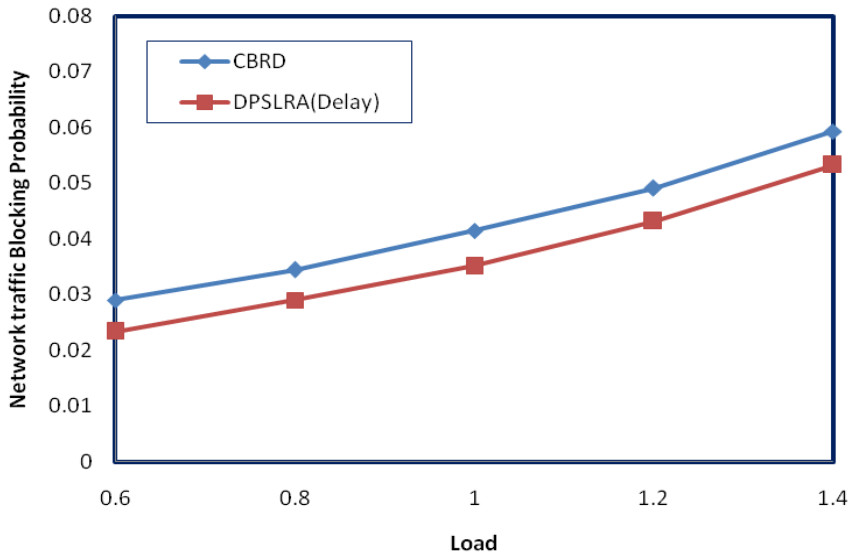
The network traffic blocking probability increases steadily as the constraint tightens over incoming network traffic, which confirms that network traffic with small delay requirements is hard to route, as expected.

5.5.2.6 Impact of network load for different topologies

Figure 5.10 shows the impact of network load in the performance of both algorithms for different topologies. In this figure the blocking probability is plotted against different various values of arrival rate for different types of network topologies. The simulation experiment starts with a small arrival rate (0.6) at which both algorithms can accept most of the arrivals as long as the average path length is small. By looking at these figures we can see that in DPSLAR delay mode performance is better than CBRD. This is due to path selection in DPSLAR delay mode. DPSLAR delay mode uses the substitution method. Therefore, the blocking probability of DPSLAR delay mode is better than CBRD in all topologies. When the arrival rate increases, CBR and DBR adapt to the change and maintain their relative performance.



(a) Torus topology



(b) RAND80 Topology (80 nodes)

Figure 5.10 : Impact of network load

5.5.2.7 Impact of heterogeneous delay constraints

In all previous simulation experiments one range of delay constraint has been used as QoS delay. The following simulation experiment aim is to study the effect of

using more than one range of delay constraint on CBRD and DPSLRA delay mode. The simulation experiment uses two types of delay constraints in order to investigate the impact of large and small constraint network traffic while having the same average holding time and average inter arrival time. The value of the delay constraints for both types is 30 for the large constraints and 15 for the small constraints, with a mean inter-arrival time of 1.The holding times for all network traffic is exponentially distributed with a mean of 2.5. Performance is measured by mixing fractions of small and large delay constraints, keeping the inter-arrival time mean fixed at 1.

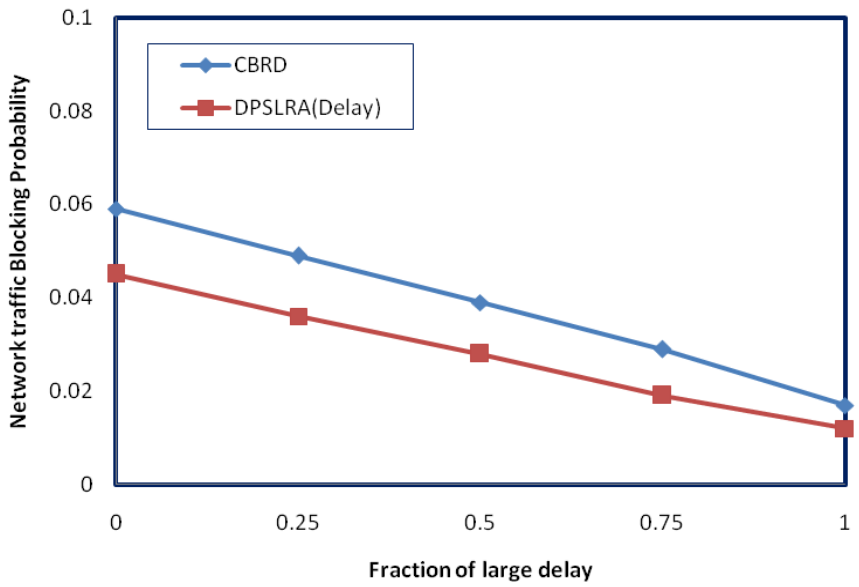


Figure 5.11 : Impact of heterogeneous delay constraint on Rand60

Figure 5.11 shows network traffic blocking probability plotted against the fraction of large delay constraints. Form this figure we can notice that the network traffic blocking probability of both algorithms is decreased as the fraction of large delay increases, which is expected in view of the fact that it is

easier for a source node to find a feasible path with a large delay constraint. DPSLAR delay mode gives the best performance. This can be expected, as DPSLAR delay mode continuously monitors the usage of candidate paths and performs path substitution where it is needed.

However, when network traffic is rejected, path credits will be decreased and an alternative path with more credits to be selected. It can also be noticed that the difference in performance remains fixed between DPSLAR delay mode and CBRD, and they both have good performance.

5.6 Chapter summary

A localized quality of service routing scheme has been developed to solve the problems associated with global quality of service routing. In localized quality of service routing, routing decisions are made based on the local view of the network quality of service state information. In this chapter we have developed two localized QoS routing algorithms: CBRD, which uses a simple crediting scheme that is increased upon network traffic acceptance and decreased upon network traffic rejection; and DPSLAR delay mode, which relies on the path substitution method in order to judge path selection for routing network traffic. We demonstrated through simulation that the two algorithms, although simple, have good performance under different network traffic loads and network topologies. Our general results suggest that localized quality of service routing

operates better if the algorithms operate directly on the QoS constraint (delay in this case) rather than indirectly (such as via a crediting scheme). Also, the path substitution appears to give promising results so methods that dynamically select the candidate path set are likely to be advantageous in performance but at a cost of increased complexity.

Chapter 6:

Disjoint Path Localized QoS Routing Algorithm (DPLRA)

6.1 Introduction

In the previous chapter we proposed the DPSLRA algorithm and CBRD algorithm to enhance the performance of localized quality of service schemes. In this chapter we propose another algorithm, which is the DPLRA algorithm that aims to maximize the end-to-end quality of service such as bandwidth. Many network applications such as multimedia applications can achieve significant performance obtained by maximizing available bandwidth. The proposed algorithm is based on the shortest edge disjoint paths.

The problem of finding disjoint paths in a network has attracted extensive research [92-97] due to its importance to many applications, such as design of integrated circuits, design of telecommunication networks and reliable

routing[98]. Paths between a given pair of source and destination nodes in a network are called link disjoint if they have no common links, and node disjoint if, apart from the source and destination nodes, they have no common nodes.

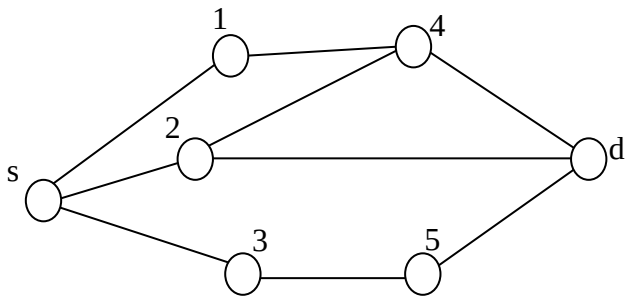
Finding shortest or several shortest paths is a common problem in communication networks. It is preferable to find two or more shortest edge disjoint paths between source node and destination node. This has many advantages as stated in [96]. The problem of finding two or more edge disjoint paths has been shown to be NP-hard [99-101].

In this chapter the focus lies on finding a localized quality of service routing scheme that uses shortest edge disjoint paths or paths that have the fewest common links between each source destination pair. A Disjoint Path Localized QoS Routing Algorithm (DPLRA) is proposed and its performance is compared to the CBR scheme.

6.2 Disjoint Path Localized QoS Routing Algorithm (DPLRA)

The idea of the DPLRA is very simple. This algorithm uses a simple method to find the set of edge disjoint paths between each source destination pair. Using edge disjoint paths in routing should maximize the bandwidth of a routing path to its full capacity. In our method of path selection process: First, we have to find the edge disjoint path set between source (s) and destination (d) as shown in figure

6.1. The network topology example is shown in figure 6.1(a) and the process of discovering is shown in figure 6.1 (b).



(a) Network topology

Process	Path	Action	Edge disjoint path set
Find path	s-1-4-d	Insert	s-1-4-d
Find path	s-2-4-d	(ignored) (4-d)	s-1-4-d
Find path	s-2-d	Insert – sort	s-2-d s-1-4-d
Find path	s-3-5-d	Insert-sort	s-2-d s-1-4-d s-3-5-d

(b) Path discovering process

Figure 6.1: Finding edge disjoint path set from s to d

As shown in figure 6.1 (b), at the beginning, we have to find the first path between source and destination by using breadth first search (BFS) and keep it; in this example the first path from source to destination {s-1-4-d}. Then using the

recursion approach we find the second path {s-2-4-d}. After that we compare it to existing path {s-1-4-d} to find if there are any common links. In this case we find that the edge (link) {4-d} is used in an existing path, therefore we have to discard this path {s-1-4-d}. We continue the process of finding new paths, so the next path is {s-2-d} that should be compared to the existing paths. The new path is accepted and the set is sorted to get the shortest path at the head of the set. The fourth path is s-3-4-d is accepted. At the end of this process we have a set of paths between source and destination. We use this set to construct the candidate path set R= $R^{\min} \bigcup R^{alt}$.

6.3 DPLRA Performance Evaluation

In the following sections, we present the simulation experiments setup that will be used to study the performance of DPLRA against the CBR scheme and widest shortest path algorithm (WSP).

6.3.1 Simulation setup (Bandwdith)

In all simulation experiments of studying the performance the DPLRA bandwidth mode, the network topology remains fixed during each experiment. Therefore, link failures are not considered. All links between network nodes are assumed to be bidirectional and have the same capacity C in each direction (C=150Mbps). The network traffic arrives to each source node according to a Poisson process with rate λ and destination nodes are selected randomly. Each

node can act as source and/or destination. Applying this uniform random selection of destinations results in a uniform traffic in regular topologies, and non-uniform traffic in random and ISP topologies, this allows us to evaluate the performance under balanced and unbalanced loads.

Recent studies [88-90] showed that the network traffic arrival process is bursty and that distributions with heavy tails, such as Weibull, yield better models. Therefore, we also study the effect of bursty traffic where network traffic inter-arrival times follow a Weibull distribution.

Network traffic bandwidths are uniformly distributed within the interval [0.1-2MB], while network traffic duration is exponentially distributed with mean $1/\mu$. To compute the offered network load references [30, 54] use equation (5.2).

The parameters for algorithms are Max_Credit=5 and Φ=1 unless otherwise stated. Blocking probabilities are calculated based on the most recent 20 connection requests.

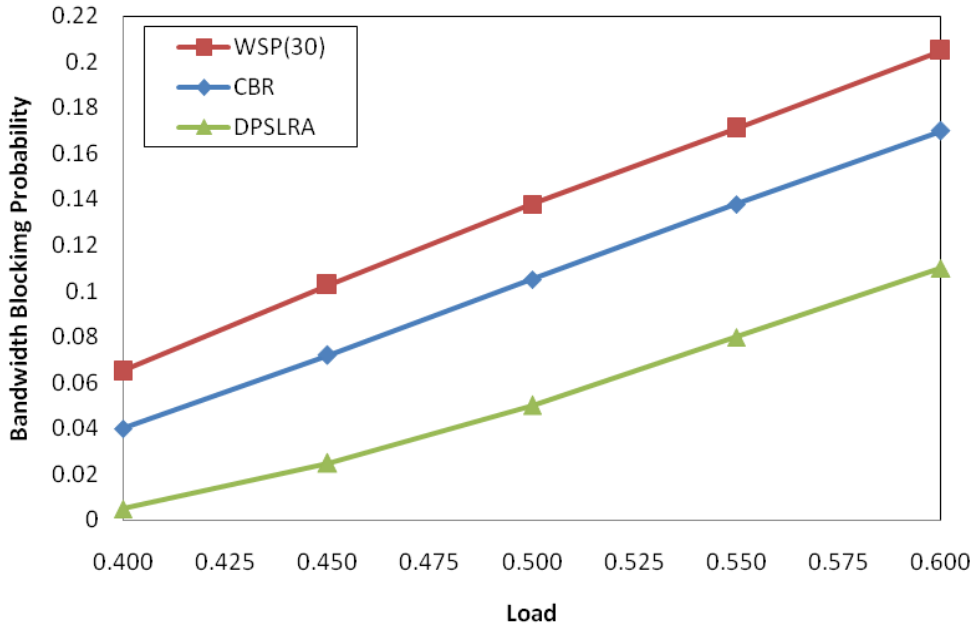
The set of candidate paths is chosen as shown in section 6.2 such that, for each source destination pair, all the paths between them are edge disjoint and whose length is at most one hop more than the minimum number of hops are included in the set. Each run simulates the arrival of 2,000,000 connection requests and the simulation results are collected after the warm-up period which represented by the first 500,000 such requests.

6.3.2 Network traffic blocking probabilities

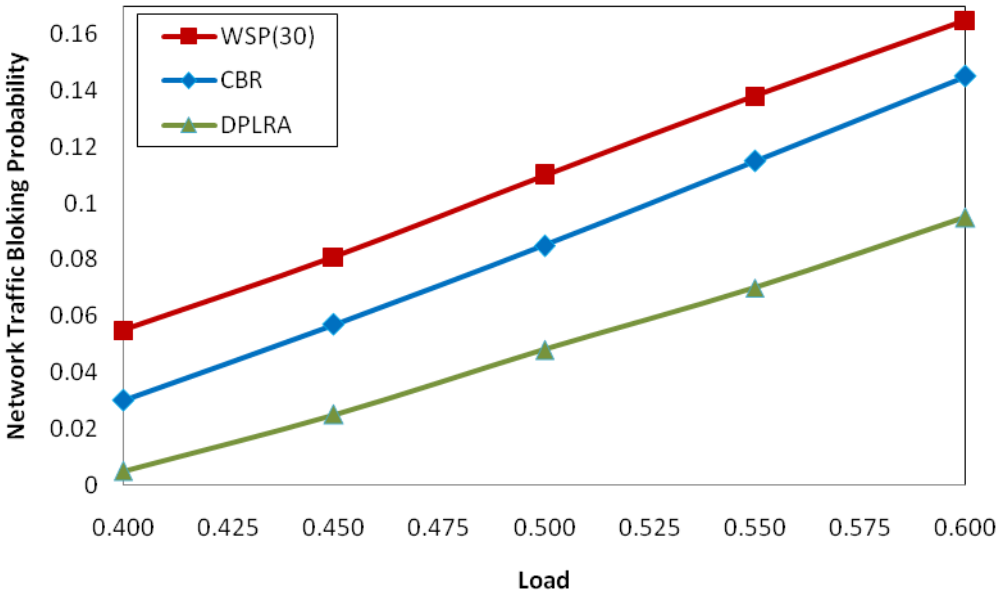
The results of a simulation experiment, which is conducted to compare the performance of the CBR, the DPLRA and the WSP in terms of network traffic

and bandwidth blocking probabilities under different load conditions, are shown in figure 6.2. In this figure, the performance of the three algorithms in terms of network traffic and bandwidth blocking probabilities under different load conditions is illustrated.

The network traffic blocking probability and bandwidth blocking probability are plotted against the offered load using the random topology RAND80. By looking at this figure we note that: first, performance varies considerably when the load increases, and the bandwidth blocking probability grows more rapidly than the network traffic blocking probability, which means that network traffics with large bandwidth requirements are hard to route, as expected. Second, the relative performance of the three algorithms is the same for network traffic and bandwidth blocking probabilities suggesting that both metrics can be used to evaluate the performance of routing algorithms especially when the bandwidth requirements are small compared to link capacities. Third, the performance of the WSP algorithm is the worst compared to the performance of other algorithms. This is likely due to the update interval of the link state. Paths are selected based on a QoS global network state information which is updated periodically and when periodic updates become stale, as they must with realistic update intervals, the performance will be affected. Fourth, the DPSLRA scheme gives the best performance under all conditions. This is due to the path selection method that selects edge disjoint paths between source node and destination node.



(a) Bandwidth blocking probabilities



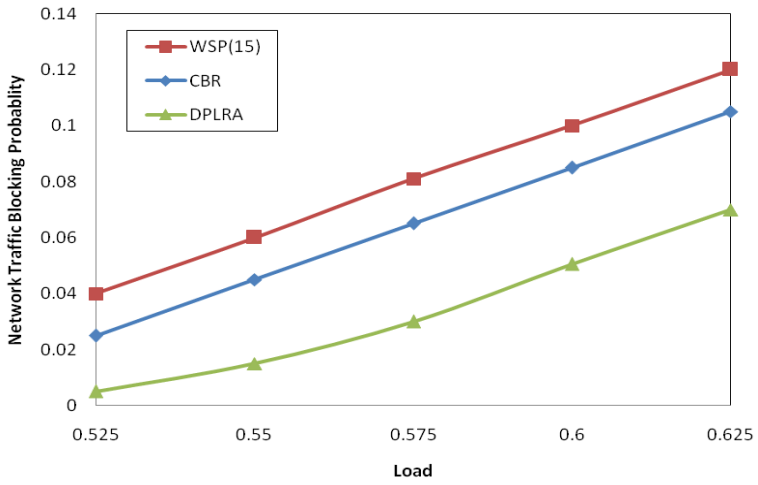
(b) Network traffic blocking probabilities

Figure 6.2: bandwidth blocking and Network traffic probabilities for RAND80

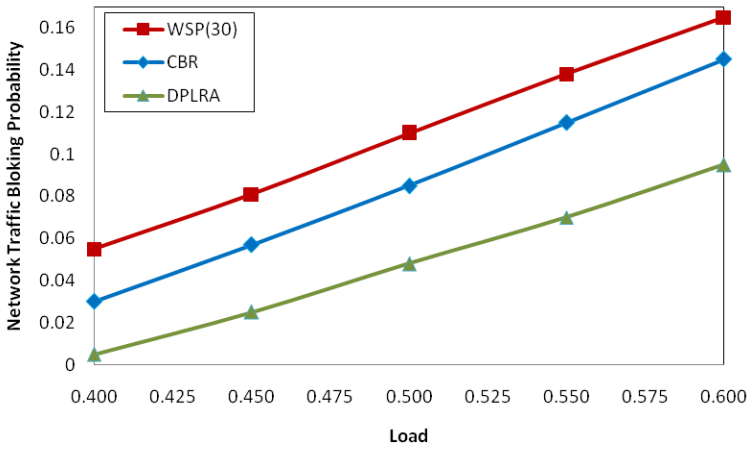
The localized quality of service routing schemes (CBR and DPLRA) almost use the same method to select the path with the maximum credits as long as it does not reject network traffic, however since credits of the selected path are updated after every network traffic routed along that path, any network traffic rejection will cause its credits to be decreased and an alternative path with more credits to be selected. The reason that makes the DPLRA outperforms the CBR, is that it uses a path selection method that uses an edge disjoint path set. The edge disjoint path set is likely to provide more residual bandwidth than the path selection method used in the CBR.

6.3.3 Impact of network topology

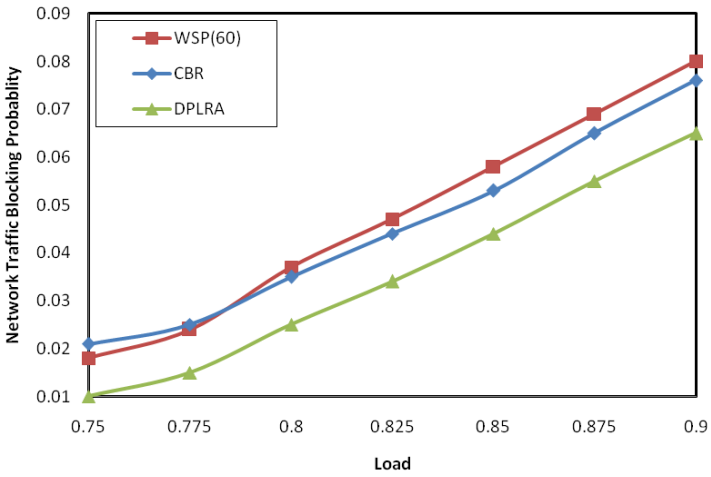
The proposed algorithm performance is simulated using different types of network topologies to demonstrate that it can perform well under across wide range of network topologies when compared with CBR and WSP.



(a) ISP topology



(b) Random topology (80 Nodes)



(c) Random topology (32 Nodes)

Figure 6.3: Impact of network

In figure 6.3 the network traffic blocking probability for the three algorithms using different network topologies is shown. From this figure we can see that the DPLRA gives excellent performance compared to the performance of the CBR algorithm and the WSP algorithm. This also is due the path selection method that is used by the DPLRA, as shown in pervious sections.

6.3.4 Impact of large bandwidth

The impact of large bandwidth on the performance the proposed algorithm under range of bandwidth will be investigated. In this case, we study the result of routing large bandwidths network traffic. This is done by using a mixed network traffic that contains two types of network traffic, small bandwidth network traffic and large bandwidth network traffic. The small bandwidth network traffic is obtained uniformly from a range [0.1-2MB] with mean b1=1.05 and [2-4MB] with mean b2=3 for large bandwidth network traffic, both types have the same flow duration.

Network traffic duration is exponentially distributed with mean equal to 190 time units for small and large bandwidth network traffic. The performance is obtained by varying the fraction of small-bandwidth network traffic f while keeping the offered load fixed at $\rho = 0.8$ by changing the arrival rate according to equation (5.3).

The network traffic blocking probability plotted as a function of the fraction of small bandwidth network traffics for RAND32 as shown in Figure 6.4.

From this figure we can notice that, as the fraction of small bandwidth network traffic increases the blocking probability decreases and again, as expected, we can see DPLRA gives the best performance.

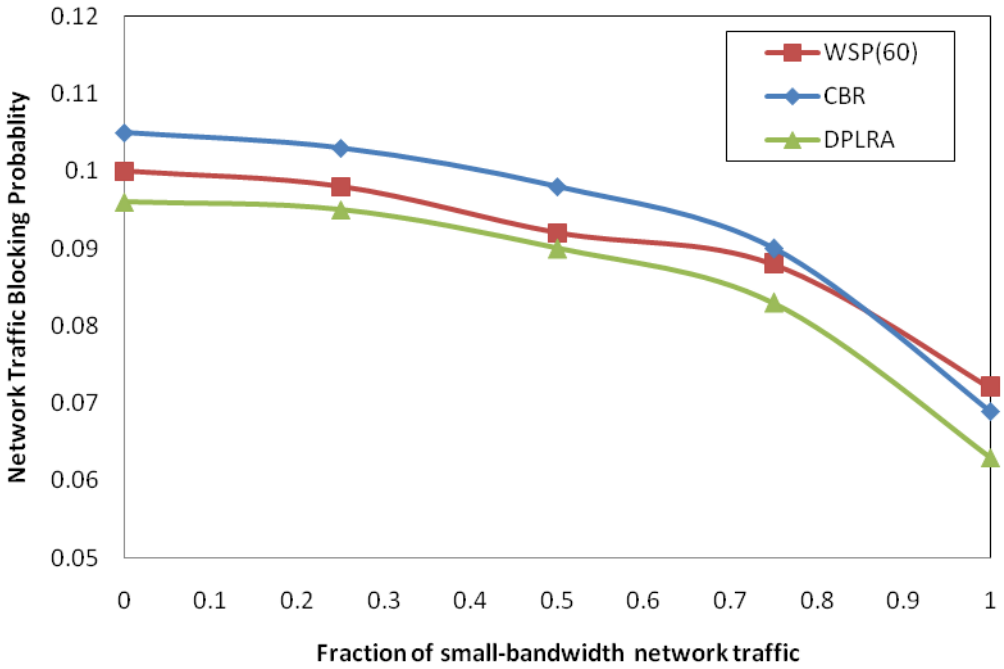


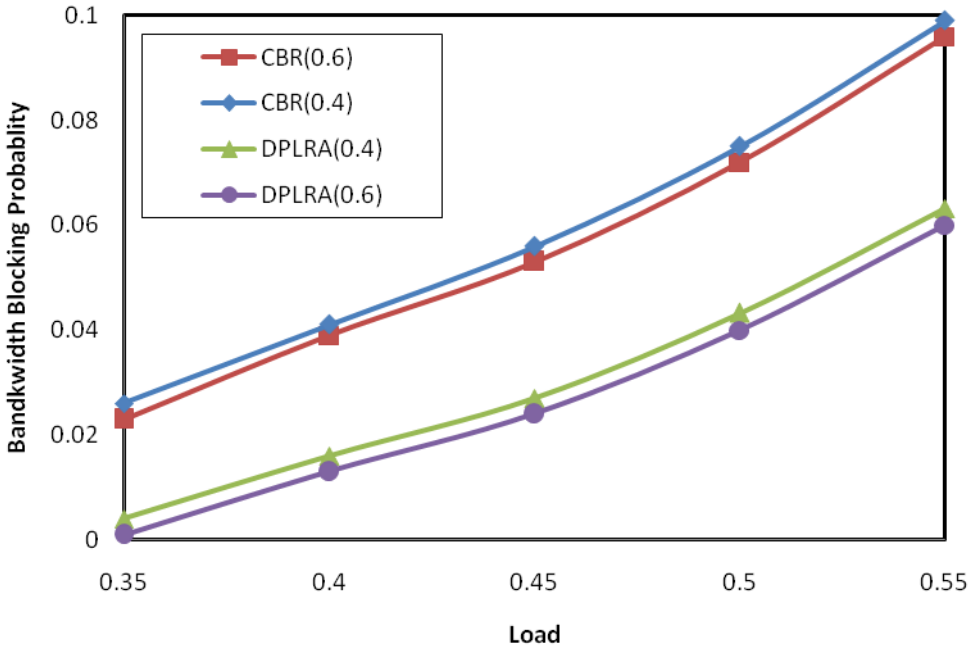
Figure 6.4: Impact of large bandwidth connection

This shows that DPLRA performs well despite the fact that routing is independent of the amount of bandwidth requested. This remains true as long as the amount of bandwidth requested is small compared to the link capacities.

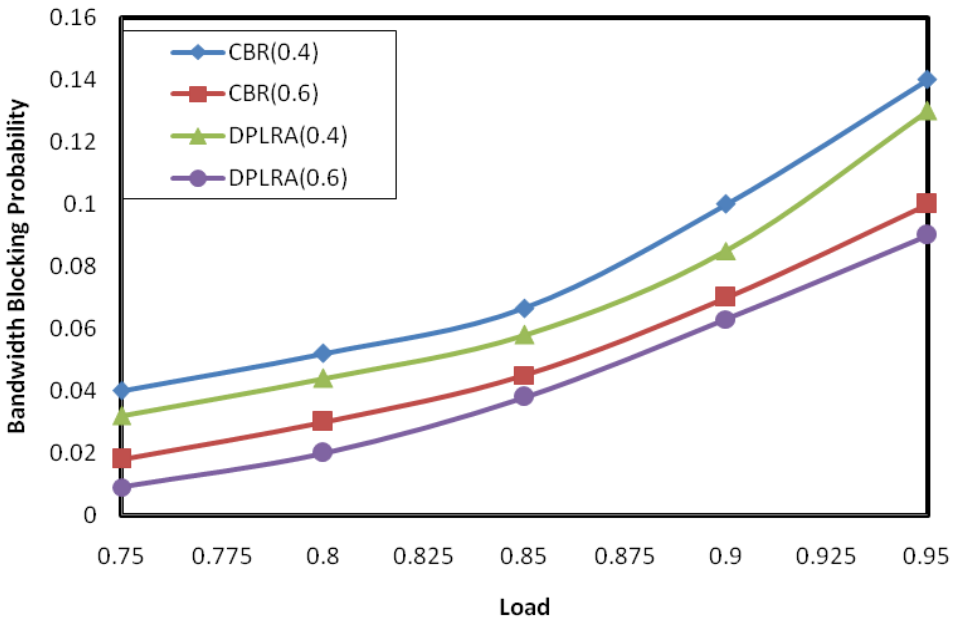
6.3.5 Impact of bursty traffic

The impact of bursty traffic on the performance of algorithms under investigation is examined by following [54, 88]. The bursty traffic is modelled using the Weibull distribution with two different values of the shape parameter of the distribution a=0.4 for large shape parameter and a=0.6 for small shape parameter where burstiness is increased with a smaller shape value.

Figure 6.5 shows the network traffic blocking probability plotted against the offered load for two random topologies, RANDOM80 and RANDOM32. From this figure we can see that the increased burstiness in the traffic arrival process results in increased blocking probability over the range of loads used.



(a) Random topology (80 Nodes)



(b) Random topology (32 Nodes)

Figure 6.5: Impact of busrty traffic

6.4 Performance Evaluation (Delay mode)

The simulation experiments setup will be presented and the performance of DPSRA in delay mode will be compared with the CBRD scheme in the following sections.

6.4.1 Simulation setup

The delay mode simulation experiments use a fixed network topology during simulation period. The network traffic form source node to destination node is generated with uniform probability. The required QoS delay is varied, ranging between 15 and 30 time units. It is assumed that all links in the topologies are bidirectional and the mean delay time for each link is exponentially distributed.

The mean of the network traffic holding time is $1/\mu$, with an exponential distribution. The network traffic's interarrival times at the source node are exponentially distributed with the mean $1/\lambda$.

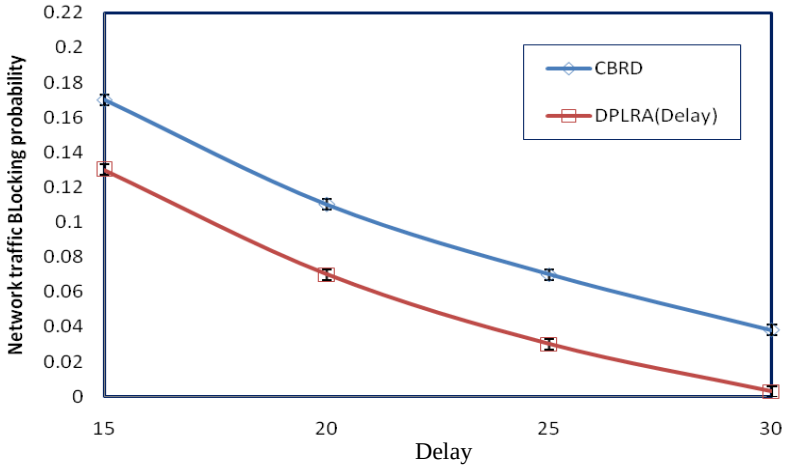
The system parameters used in the simulation are Max_Credit=5 and Φ=1. Blocking probabilities are calculated based on the most recent 20 flows for both algorithms. For CBR, candidate paths between each source-destination pair in a network topology are chosen, so we include minimum hop and minimum hop +1 in the set to get the required number of candidate paths between each pair. For, DPLRA, candidate paths between each source-destination pair in network topology are chosen as shown in section 6.2. All experiments simulate 2,000,000

connection requests and results are collected after the first 500,000 connection requests (used for simulation warm-up).

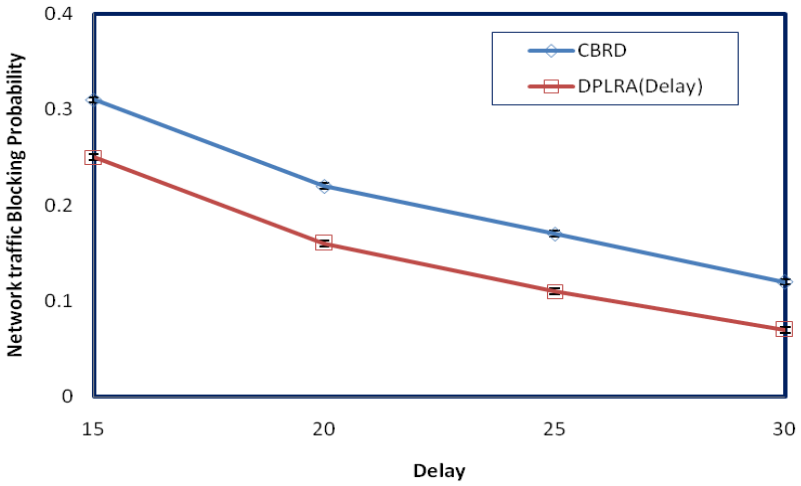
Network traffic blocking probability was used to measure the performance of the algorithms. Network traffic will be rejected when the path does not satisfy the delay constraint and/or jeopardizes the constraint of an existing flow. The blocking probability is defined as in equation 5.5.

6.4.2 Network traffic blocking probabilities

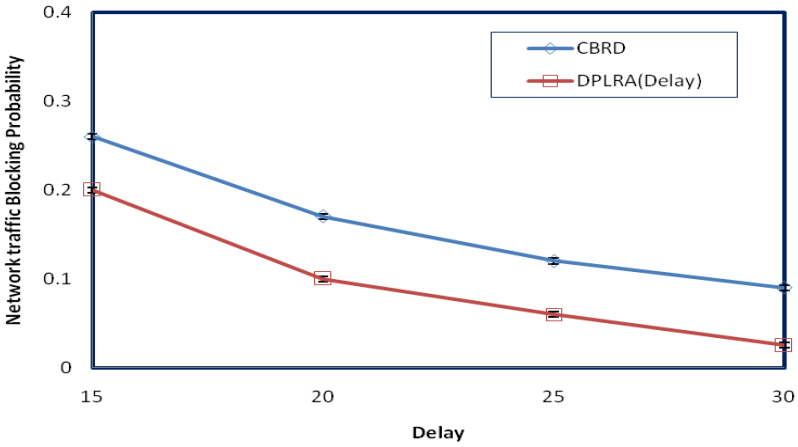
Figure 6.6 shows the network traffic blocking probability plotted against various delay constraints for different types of network topologies. It can be noted that all algorithms satisfy most requests under large delay constraint (constraint 30), which can be expected as the probability of finding a path that satisfies a large delay constraint is high and most network traffic will be accepted. In figure 6.6 (a) (ISP topology), both algorithms give good performance with the same constraint. This is due to the fact that ISP topology has the smallest average path length. However, performance under some constraints may be significantly degraded based on the average path length in the topology. This can be noticed in figure 6.6. (b), with Lattice topology giving the worst performance compared with the other topology under a delay constraint of 30. This is because the Lattice topology has the largest average path length.



(a) ISP topology



(b) Lattice Topology



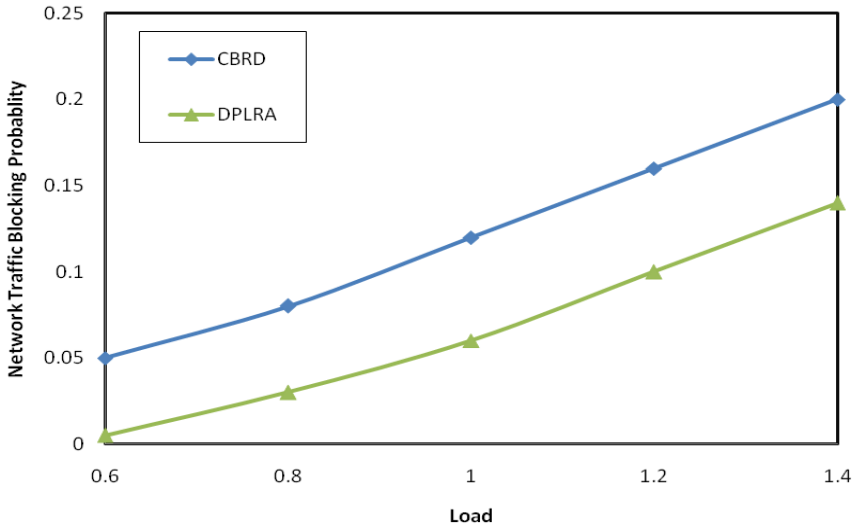
(c) Torus topology

Figure 6.6: Network traffic blocking probability under various network topologies and various delays

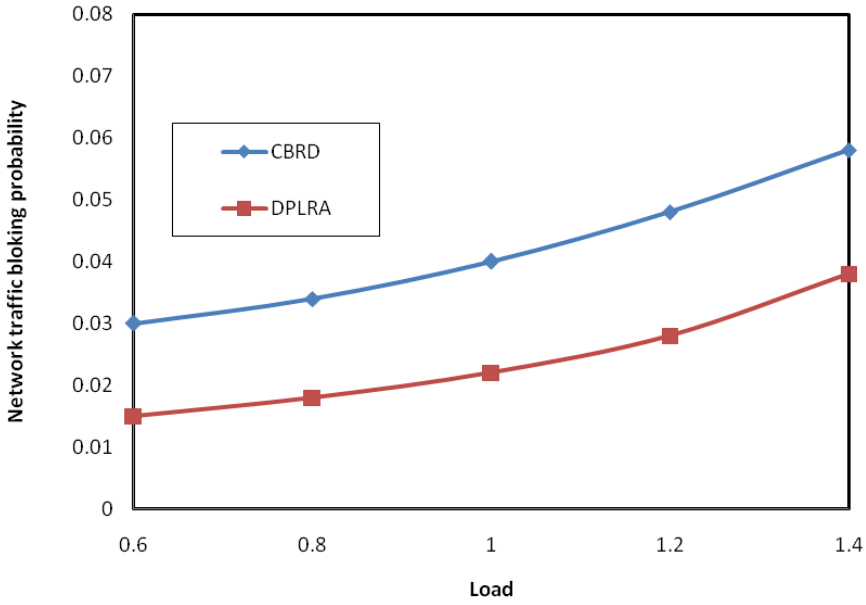
The network traffic blocking probability increases as the constraint tightens over incoming network traffic, which confirms that network traffic with small delay requirements are hard to route, as expected. But, DPLRA gives better performance than CBRD again due to its path selection method.

6.4.3 Impact of network topology

Figure 6.7 shows the impact of network topology in the performance of both algorithms. The blocking probability is plotted against various different values of arrival rate for different types of network topologies. Both algorithms can accept most of the arrivals with a small arrival rate (0.6) as long as the average path length is small. From these figures we can notice that DPLRA delay mode gives superb performance compared to the CBRD performance. This is due to path selection in DPLRA delay mode as mentioned in earlier sections. Therefore, the blocking probability of DPLRA delay mode is better than CBRD in all topologies.



(a) Torus topology



(b) RAND80 Topology (80 nodes)

Figure 6.7: impact of network topology

6.4.4 Impact of heterogeneous delay constraint

The aim of the following simulation experiment is to study the effect of using more than one delay constraint on CBRD and DPLRA delay mode. The simulation experiment uses two delay constraints in order to study the impact

of large and small constraint network traffic while having the same average holding time and average inter arrival time. The value of the delay constraints is 30 for the large constraints and 15 for the small constraints, with a mean inter-arrival time of 1. The holding times for all network traffics are exponentially distributed with a mean of 2.5. Performance is measured by mixing fractions of small and large delay constraints, keeping the inter-arrival time mean fixed at 1.

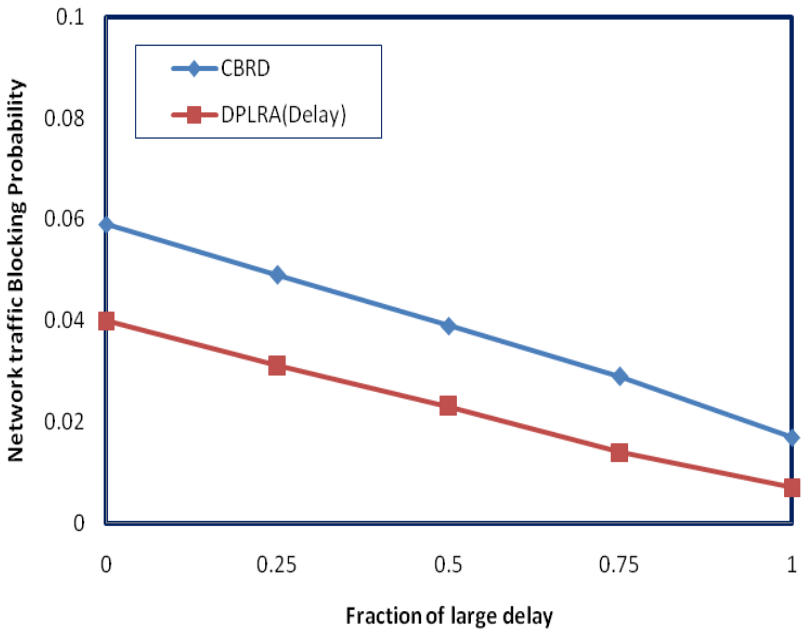


Figure 6.8: Impact of heterogeneous delay constraint on Rand60

Figure 6.8 shows network traffic blocking probability plotted against the fraction of large delay constraints. Form this figure we can notice that the network traffic blocking probability of the both algorithms is decreased as the fraction of large delay increases, which is expected in view of the fact that it is

easier for a source node to find a feasible path with a large delay constraint. DPLRA delay mode gives the best performance. This can be expected, as DPLRA delay mode uses shortest edge disjoint paths that maximize the path effectiveness and efficiency.

6.5 Chapter summary

The candidate path selection method has significant impact on the performance of routing algorithms. In this chapter we have developed a path selection method that improves the performance of localized QoS routing algorithms. We implement the developed path selection method in the DPLRA. Then we compare the DPLRA performance with the CBR algorithm performance, we found that DPLRA outperforms the CBR. This is demonstrated through simulation of the two algorithms under different network traffic loads and network topologies. Simulation results suggest that localized quality of service routing schemes should use a good path selection mechanism for the candidate paths that, as far as possible select the shortest path with fewest common links. (preferably disjoint path) Also, simulation results showed that localized QoS routing can be successfully employed in a network with multi-constraint delay requirements or heterogeneous traffic.

Chapter 7:

Hierarchical Localized QoS Routing Algorithm (HLRA)

7.1 Introduction

Most current routing algorithms suffer from scalability problems as the size of the network increases. The overhead associated with maintaining and distributing the global state of the network, and the routing computation overhead create major challenges in designing QoS routing algorithms for large networks. In the first part of this thesis, we proposed an effective and simple localised approach to routing that proved a viable alternative in overcoming these challenges. On one hand, the approach relies on statistical information collected locally which eliminates the need for a global state. On the other hand, path computation involves a simple selection of the desired path from a set of the candidate paths R, resulting in a very low time complexity. However these advantages remain specific to localised approaches. Therefore, in the rest of this thesis, we turn our

attention into investigating general techniques that can be utilized to enhance the scalability of QoS routing algorithms in large networks.

The common problem in large networks is the growth in size of the network state information. Hierarchical routing is a widely used technique that has been used to deal with the scalability problem in large networks [102-104]. It uses an aggregated global state described in chapter two. In this approach, a network is divided into sub-networks called autonomous systems (AS) and each node maintains detailed state information about nodes in the same (AS) and aggregated state information about other groups. Nodes in the same (AS) run the same routing intra-AS routing protocol. In each (AS) there is one or more gateway-node which is responsible for connecting to other nodes outside the (AS) by running an inter-AS routing protocol.

When a node receives a connection, it uses source routing to construct a path from source node to destination node using a source routing algorithm. Since the state information kept at each node is an aggregated global state, some nodes of the computed path may be logical nodes representing groups. A control message is sent along the path to establish the connection. When the control message is received by a border node that represents a logical group, it uses a source routing algorithm to expand the path through that group.

Hierarchical routing uses source routing and the path computations are distributed over many nodes also. Hence, it combines the advantages of source routing and distributed routing. The size of the state maintained at each node is logarithmic in

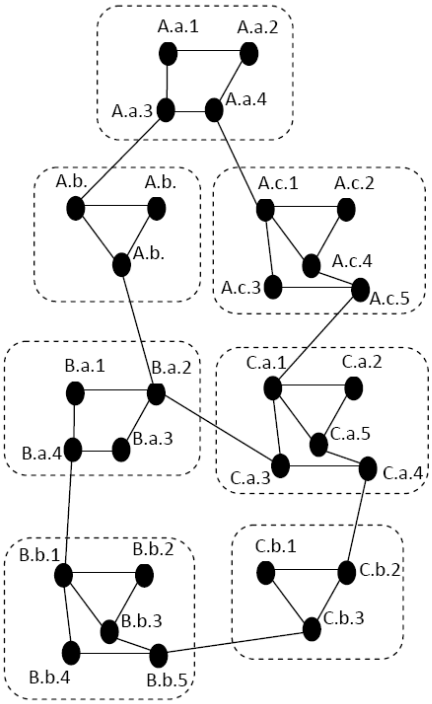
the size of the complete global state, which means that hierarchical routing scales very well due to the reduction of nodal storage, state exchange overhead and computational effort.

The major drawback of Hierarchical routing is the imprecision in state information introduced by aggregation, which has a significant negative impact on the routing algorithm [14, 105, 106]. As the number of levels increases, the imprecision introduced by aggregation increases too.
Moreover, it is very difficult to determine the best approach to represent a logical group. Single node representation, star representation and mesh representation are common approaches in the literature and every approach has its pros and cons [107].

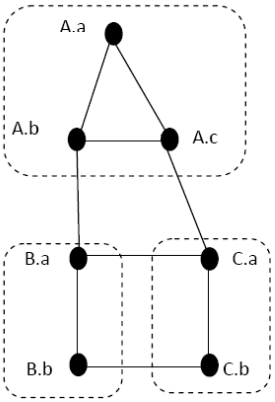
7.2 Hierarchical Model aggregated network state

This state represents the hierarchical structure of the network. Hierarchical models are often used to reduce the size of the global state by having each node store more detailed information about nodes that are close to it and less information about the nodes far from it.

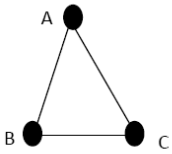
Figure 7.1 shows the hierarchical model used by ATM networks [7, 10]. In Figure 7.1.a, nodes are clustered into the first level group. Nodes with links crossing clusters are called border nodes. In Figure 7.1.b, each group is represented by a logical node.



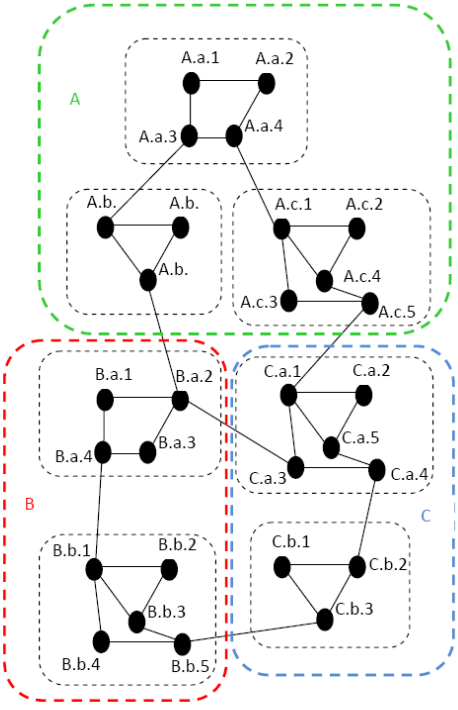
(a) Physical network



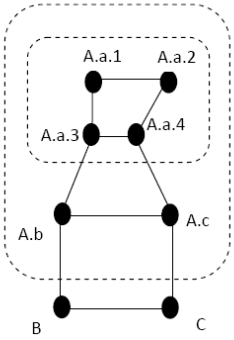
(b) First level abstraction



(c) Second level abstraction



(d) Clustering



(e) The network image viewed by node A.a.1

Figure 7.1 : Hierarchal Model used by ATM networks

A physical node in each group is selected to act on behalf of the logical node and store higher-level state information. Links that connect logical nodes are called logical links. Logical nodes may be further clustered further to form higher-level logical nodes as shown in Figure 7.1.c. On each level, the nodes in the same group are called the children of the logical node representing the group; the logical node itself is called the parent.

It should be noted that using a single logical node to represent a group is not the only way of doing topology aggregation. Each physical node maintains an aggregated network image. The image maintained by node A.a.1 is shown in Figure 7.1.e. As the network topology is aggregated, the state information is aggregated too and the state of each logical link is the combination of the states of many lower-level links. This reduction of state information size does not come for free as the aggregation introduces imprecision in the state information. So there is a trade-off between the size of the aggregated state and the imprecision introduced by this aggregation.

Although, many such techniques with different goals have been proposed to deal with the inherent scalability of QoS routing [108], our focus will be on the studying of hierarchical routing to scale routing algorithms in terms of computational effort and reducing message overhead.

7.3 Scalability Techniques for QoS Routing

To understand the relationship between hierarchical routing studied in this chapter and other techniques for scaling QoS routing algorithms we introduce the taxonomy shown in Figure 7.2.

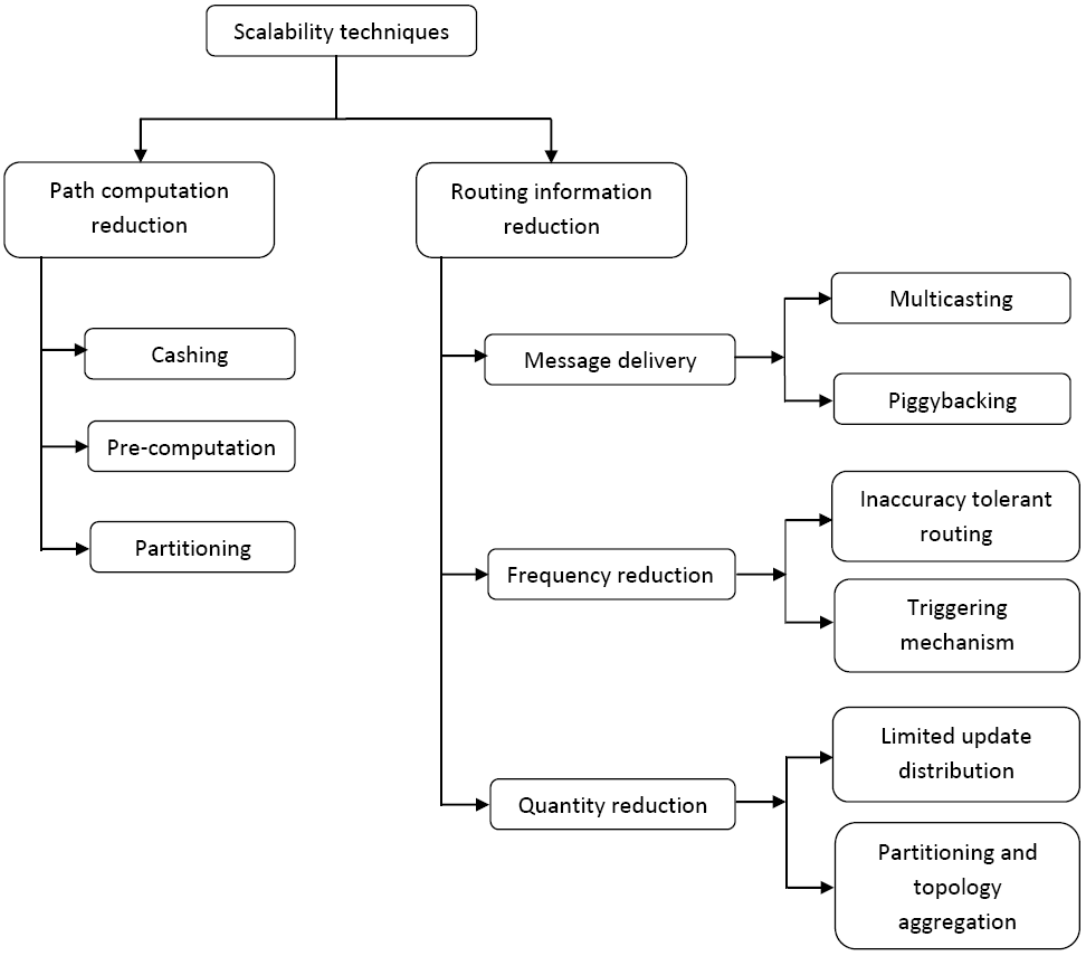


Figure 7.2 : Taxonomy of Scalability techniques. (Adapted from [106])

As we can see, techniques are classified according to the two major overheads of QoS routing: path computation reduction and routing information reduction. The first type addresses the issue of how to reduce the computational effort in finding a feasible path that satisfies the QoS requirements given that routing information is already available. Stochastic partitioning belongs to this type. The second type is about reducing the routing information needed to be maintained and exchanged to perform path computation.

For the sake of completeness, we give a brief description of each type with some examples.

7.4 Routing information reduction

Routing information reduction techniques can be classified based on the focus of reduction. Quantity reduction methods focus on reducing the number and the size of routing information by addressing the issues of “what information should be maintained and exchanged” and “where this information should be delivered to?”. The well known Hierarchical routing [102, 109] used in the Internet is an example of a quantity reduction technique. This method relies on partitioning the network into sub-networks where each router maintains a detailed state of its sub-network and an aggregate state of other sub-networks with the aim of reducing the size of routing tables at each router. The topology aggregation used in ATM networks is another example of quantity reduction methods. Another approach addresses the issue of “where routing information should be delivered to?” by limiting the scope

of distributing state information to be within a certain hopcount limit where the information is most needed [110].

Frequency reduction techniques focus on the issue of "how frequent the routing information should be distributed". This is achieved by either some triggering mechanisms such as time-based triggers or change-based triggers [111, 112] or by designing routing algorithms that tolerate infrequent state updates [110, 113].
Message delivery reduction methods focus on how to deliver the routing information efficiently. Common approaches include using multicast techniques to distribute routing information and network state instead of flooding and/or piggybacking routing information in control messages such as flow signalling messages [87].

7.5 Path computation reduction

The three main techniques for scaling QoS routing algorithms in terms of computational effort are caching, pre-computation, and partitioning. Caching is widely used as a speed up technique in many fields. In the context of QoS routing, caching involves reusing paths that are computed on an "on-demand" basis by storing these paths and their related information for future use. Upon flow arrival, the cache is consulted to see if there is a cached path that can support the arriving flow or not. If so, the cached path will be used to route the new flow thus avoiding computing a new path, otherwise on-demand computation is required and a cache "miss" occurs. Major issues related to any caching scheme need to be addressed

here, such as when and how to update/invalidate cached paths and what replacement policy should be used. A discussion of many issues relating to this technique in the context of QoS routing can be found in [29, 114].

7.6 Hierarchical Localized QoS Routing Algorithm (HLRA)

A Hierarchical Localized QoS Routing Algorithm (HLRA) is proposed to improve the performance of QoS routing algorithms in large computer networks. The basic idea of the proposed algorithm is to apply the localized QoS approach in hierarchical networks.

Applying the localized QoS approach in hierarchical networks should reduce the computations, overheads and storage associated with the candidate paths compared to a flat network approaches retaining the inherent advantages of localized routing over the global techniques. The pseudo code of the proposed algorithm is shown in figure 7.3.

```
Initialize

set   P.credits = Max_Credit, ∀   P ∈ R
HLRA(Max_Credit)
1       if  distination_node ∈ AS
2             route flow to distination_node
3       else
4             route flow to Source.AS.border_node
5             route flow from  Source.AS.boarder_node to distination_node.AS.border_node
6             route flow from  distination_node.AS.border_node to distination_node
7       if  flow accepted
8             UpdateBlockingProbability(P,1)
9             amount = 1 - P.getBlockingProbability()
10             P.credits = min(P.credits + amount, Max_Credit)
11      else
12             UpdateBlockingProbability(P,0)
13             amount = P.getBlockingProbability()
14             P.credits = max(P.credits - amount,0)
```

Figure 7.3 : The pseudo-code of the Hierarchical Localized QoS Routing Algorithm

In the proposed Hierarchical Localized QoS Routing Algorithm shown in figure 7.3, every node inside AS has to maintain a set of candidate paths to all other nodes in the same AS and each border node has to maintain set of candidate paths to other border nodes in the network, when a node receives a connection request, the algorithm will check if the destination node is in the AS (1) by consulting its candidate path set. If the destination node is in the same AS, it will route the network traffic to that node by using a localized QoS algorithm described in chapter 6. Otherwise, the destination node will be outside the AS which means that the connection request will be forwarded to the AS border node and then to the border node of the AS that contains the destination node (4-6). The (HLRA)

will collect the information regarding the network acceptance or rejection to be used in monitoring that condition the candidate path set (7-14) as described in the localized QoS routing schemes in pervious chapters.

7.7 HLRA performance evaluation

In this section, the HLRA performance will be tested under various conditions and the simulation setup environment and results will be presented.

7.7.1 Simulation setup

The simulation of the proposed algorithm is done by creating a hierarchical structure. The hierarchical structure is composed of 10 AS. Each AS contains a 32 nodes and 108 links as shown in figure 7.4. The links inside the AS have capacity C (C=150MB) and the links between sub-networks (ASs) have capacity (HC=500MB). All links are bidirectional. The selection of source and destination nodes is a random process with network traffic generated according to a Poisson process with rate λ. Each node can act as source and/or destination. Using this uniform random selection of destinations results in a uniform traffic in regular topologies, and non-uniform traffic in random topologies, this allows us to evaluate the performance under balanced and unbalanced loads.

According to [88-90] the network traffic arrival process is bursty and distributions with heavy tails, such as Weibull, yield more realistic models. Therefore, we also study the effect of bursty traffic where network traffic inter-

arrival times follow a Weibull distribution. Network traffics bandwidths are uniformly distributed within the interval [0.1-2MB], while network traffic duration is exponentially distributed with mean $1/\mu$. The offered network load is computed using equation (5.2).

All localized QoS routing schemes use the required parameters as shown in previous chapters. Blocking probabilities are calculated based on the most recent 20 connection requests.

The set of candidate paths is chosen as appropriate for each localized QoS scheme used. Each run simulates the arrival of 2,000,000 connection requests and the simulation results are collected after the warm-up period which is represented by the first 500,000 connection requests.

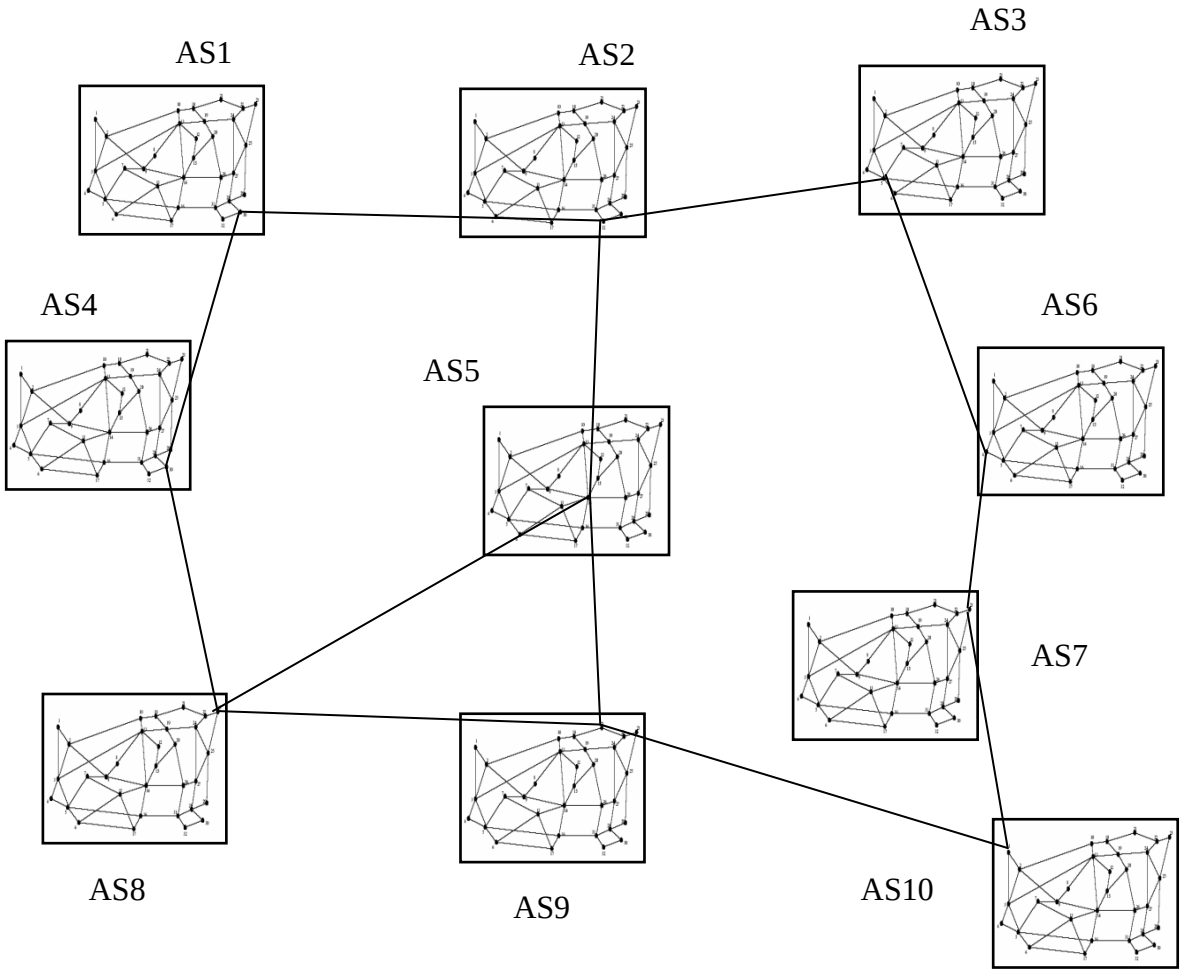


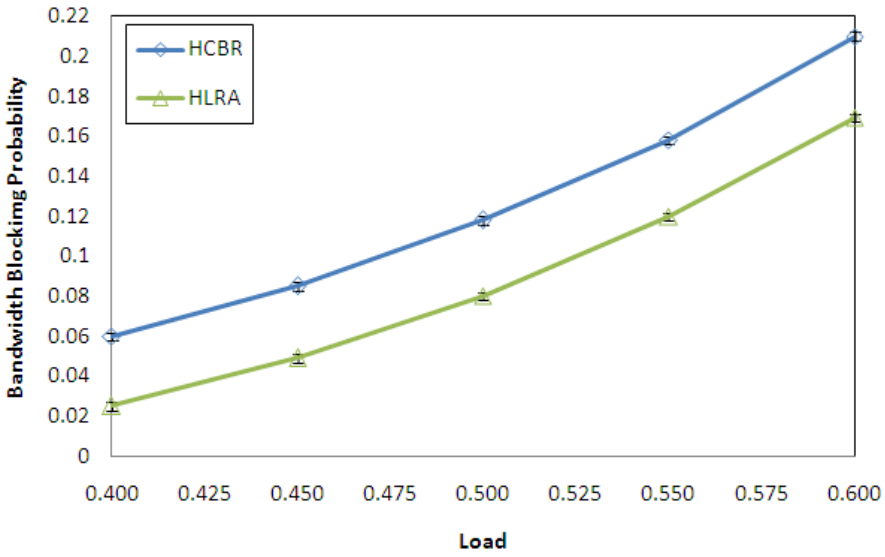
Figure 7.4 : Hierarchical topology

7.7.2 Network traffic blocking probabilities

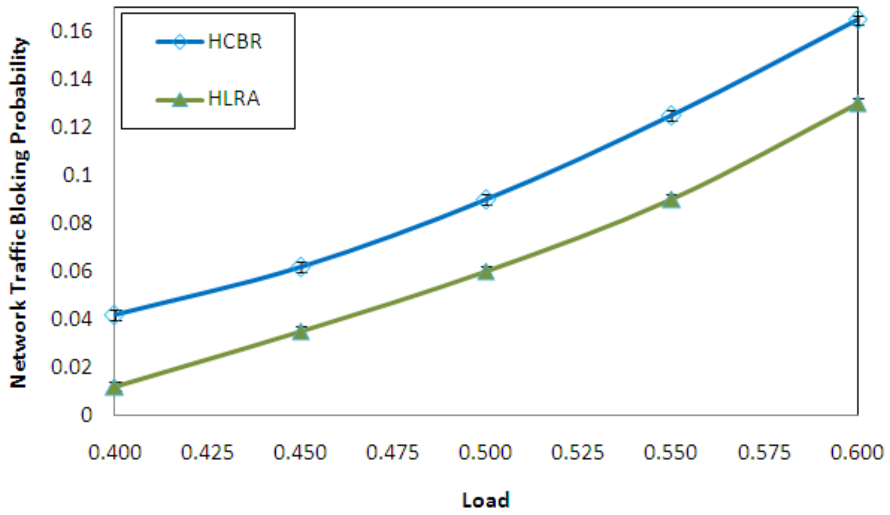
This simulation experiment is concerned with the performance of the HLRA and the HCBR in terms of network traffic and bandwidth blocking probabilities under different load conditions. Figure 7.4 shows the performance of the two algorithms in terms of network traffic and bandwidth blocking probabilities under different load conditions. This experiment is repeated 10 times and the

95% confidence were calculated. The confidence intervals are sufficiently small to be ignored as shown in figure 7.5.

The overall network traffic and bandwidth blocking probabilities are plotted against the offered load using the random topology RAND320. We note that: First, the performance of the two algorithms is the same for network traffic and bandwidth blocking probabilities suggesting that both metrics can be used to evaluate the performance of routing algorithms especially when the bandwidth requirements are small compared to link capacities. Second, the HLRA scheme gives the best performance under all conditions.



(a) Bandwidth blocking probabilities



(b) Network traffic blocking probabilities

Figure 7.5 : bandwidth blocking and Network traffic probabilities for RAND320

There are a number of reasons for the above performance results: the HCBR and the HLRA algorithms almost share the same mechanism to select a path with the maximum credits as long as it does not reject flows, however since credits of the selected path are updated after every network traffic routed along that path, any

flow rejection will cause its credits to be decreased and an alternative path with more credits to be selected. The advantage of the HLRA, that makes it outperform the HCBR, is that it uses disjoint paths. Using disjoint paths clearly increases the bandwidth availability. Applying both algorithms to large networks such as the Internet makes the localized quality of service approach scaling to large networks.

7.7.3 Impact of bursty traffic

As shown in [54, 88] we model the bursty traffic using the Weibull distribution with two different values of the shape parameter of the distribution a=0.4 for the large shape parameter and a=0.6 for the small shape parameter where burstiness is increased with a smaller shape value. Figure 7.6 shows the network traffic blocking probability plotted against the offered load for two random topologies, RAN320, as we can see in the figure increased burstiness in the traffic arrival process results in increased blocking probability over the range of loads used.

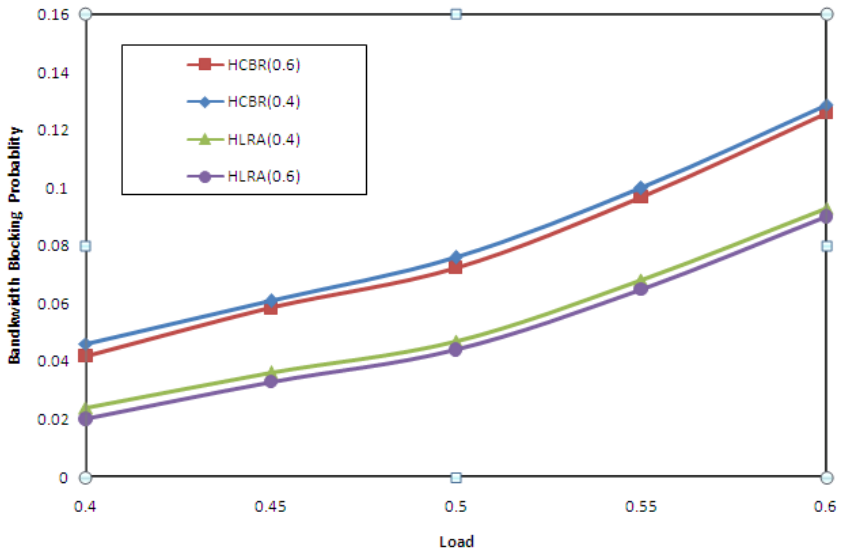


Figure 7.6 : Impact of bursty traffic

7.8 Chapter summary

In this chapter, we have extended our localized QoS routing schemes to provide hierarchical QoS routing across large networks that are consists of several sub-networks. We proposed the HLRA and HCBR to enhance the performance and scalability of QoS routing algorithms in large networks. We have shown through simulation experiments that the proposed algorithms demonstrate acceptable and good performance in a hierarchical network topology structure. This proves that the localized quality of service schemes can be used in large networks such as the Internet. The usage of the disjoint or least common links technique improves the performance of the localized quality of service routing schemes as shown in chapter 6. Therefore, combining these techniques yield a significant impact in the performance of localized QoS routing in large networks.

Chapter 8:

Conclusions and future work:

8.1 Summary and Conclusions

The scalability of the QoS routing problem in large networks is due to the following: overhead associated with maintaining and distributing the global network QoS state information, and the overhead imposed by path computation. Any effective scaling approach should target at least one of these major overheads.

In chapter 4, we conducted an extensive review of issues related to communication networks representation and simulation. This includes simulation of routing algorithms, graph models, simulation design and simulation implementation. Although, there are many ready made simulation packages available, we preferred to build our simulation using Java because of the specialized nature of the routing algorithms to be evaluated.

In chapter 5, we demonstrated how localised QoS routing in general and the proposed DPSLRA in particular could be considered as alternative scalable approaches to the QoS routing problem. The DPSLRA uses the substitution method in order to choose a path to convey network traffic. The simulation results suggested that this technique is could improve the performance of localised QoS routing schemes that use either bandwidth or delay as a routing constraint.

In chapter 6, we continued to focus on localized QoS routing schemes and proposed the DPLRA. This algorithm depends on the path selection method. The path selection method used in the DPLRA selects the shortest disjoint paths or paths that have the fewest common links between each source destination pair. Simulation's results showed that using this technique improves the performance of localised QoS routing schemes. The performance improvement is due to the increase in the amount of bandwidth availability made by employing disjoint paths.

In chapter 7, we focused in the performance of localized schemes in large networks. We introduced HLRA and HCBR and demonstrated how to implement a hierarchical localized QoS routing in large networks such as the Internet. This proves that the hierarchical localized QoS routing approach can operates and scale to large networks.

It has been demonstrated throughout the thesis that performance of any QoS routing algorithm depends on many factors such as network topology, traffic characteristics, load conditions and the strictness of the QoS constraints. Hence all

these factors must be taken into account when evaluating the performance of any QoS routing algorithm in order to give fair conclusions.

8.2 Future Works

The following are possible directions for future work:

- DPSLRA and DPLRA assume that the QoS requirements can be translated in terms of single metric such as bandwidth or delay. It would be interesting to investigate ways to extend this approach to allow the multi-metrics of different types.
- Selecting candidate paths clearly has a significant impact on how localised QoS routing algorithms perform, and this needs to be investigated further in the context of the proposed localised algorithms.
- Implementing the localised QoS routing approach in the case of QoS multicast routing is another important area. To the best of our knowledge, no multicast algorithm has been developed based on this localised approach.
- Investigating the possibility of implementing the proposed localised QoS routing schemes in MANETs might be another area to look at.
- Investigating the possibility of using AI techniques to improve path selection mechanism could also prove useful.
- Finding an easy way to calculate shortest disjoint paths.

References:

[1] **J. Moy, "OSPF version 2," *Internet Engineering Task Force,* April 1998.**

[2] **O. Y. a. S. Fahmy, "Constraint-Based Routing in the Internet: Basic Principles and Recent Research," *IEEE Communications Surveys & Tutorials,* vol. 5, pp. 2-13, 3rd Quarter 2003.**

[3] **J. M. D. Awduche, J. Agogbua, M. O'Dell and J. McMaus, "Requirements for Traffic Engineering over MPLS," *Internet Draft, <draftietf-mpls-traffic-eng.00.txt>,* October 1998.**

[4] **R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, "Resource ReSerVation Protocol (RVSP) - Version 1 Functional Specification," *IETF RFC 2205,* 1997.**

[5] **E. Rosen, A. Viswanathan, and R. Callon, "Multi-protocol label switching architecture," *RFC 3031,* 2001.**

[7] **R. Braden, D. Clark, and S. Shenker, "Integrated Services in the Internet Architecture: an Overview," *RFC 1633,* June 1994.**

[8] **S. Blake, "An Architecture for Differentiated Services," *IETF RFC 2475,* December 1998.**

[9] **S. Nelakuditi, Z. Zhi-Li, R. P. Tsang, and D. H. C. Du, "Adaptive proportional routing: a localized QoS routing approach," *Networking, IEEE/ACM Transactions on,* vol. 10, pp. 790-804, 2002.**

[10] **S. H. Alabbad and M. E. Woodward, "Localized Credit Based QoS Routing : Performance Evaluation Using," in *Simulation Symposium, 2006. 39th Annual,* 2006, pp. 42-49.**

[11] **W. C. Lee, M. G. Hluchyi, and P. A. Humblet, "Routing subject to quality of service constraints in integrated communication networks," *Network, IEEE,* vol. 9, pp. 46-55, 1995.**

[12] **R. Guerin and A. Orda, "QoS based routing in networks with inaccurate information: theory and algorithms," in *INFOCOM '97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE,* 1997.**

[13] **G. Apostolopoulos, R. Guerin, S. Kamat, and S. K. Tripathi, "Improving QoS routing performance under inaccurate link state information," *Proc. 16th Int. Teletraffic Congr,* 1999.**

[14] **C. Shigang and K. Nahrstedt, "An overview of quality of service routing for next-generation high-speed networks: problems and solutions," *Network, IEEE,* vol. 12, pp. 64-79, 1998.**

[15] **D. H. Lorenz and A. Orda, "QoS routing in networks with uncertain parameters," *Networking, IEEE/ACM Transactions on,* vol. 6, pp. 768-778, 1998.**

[16] **R. G. G. Apostolopoulos, S. Kamat, and S. K. Tripathi, "Quality of service based routing: a performance perspective," *Proc. ACM SIGCOMM,* 1998.**

[17] **R. A. Guerin, A. Orda, and D. Williams, "QoS routing mechanisms and OSPF extensions," in *Global Telecommunications Conference, 1997. GLOBECOM '97., IEEE*, 1997, pp. 1903-1908 vol.3.**

[18] **W. Zheng and J. Crowcroft, "Quality-of-service routing for supporting multimedia applications," *Selected Areas in Communications, IEEE Journal on*, vol. 14, pp. 1228-1234, 1996.**

[19] **R. Vogel, R. G. Herrtwich, W. Kalfa, H. Wittig, and L. C. Wolf, "QoS-based routing of multimedia streams in computer networks," *Selected Areas in Communications, IEEE Journal on*, vol. 14, pp. 1235-1244, 1996.**

[20] **A. V. G. Boris V. Cherkassky, Tomasz Radzik "Shortest Paths Algorithms: Theory And Experimental Evaluation," *ACM-SIAM Symposium on Discrete Algorithms (A Conference on Theoretical and Experimental Analysis of Discrete Algorithms)*, 1994.**

[21] **E. Crawley, R. Nair, B. Rajagopalan, and H. Sandick, "A Framework for QoSbased Routing in the Internet," *IETF RFC 2386*, 1998.**

[22] **I. S. Organization, "Intra-Domain IS-IS Routing Protocol," *ISO/IEC/JTC1/SC6 WG2 N323*, 1989.**

[23] **C. Hedrick, "Routing Information Protocol," *RFC 1058, Network Information Center, SRI Internation, Menlo Park, CA*, 1988.**

[24] **C. L. H. Rutgers, "An introduction to IGRP," *The State University of New Jersey, Center for Computers and Information Services, Laboratory for Computer Science Research*, 1991.**

[25] **E. Dijkstra, "Note on two problems in connation with graphs," *Numersiche Mathematik,* vol. 1, pp. 269-271, 1959.**

[26] **Gallager, "Data Networks," *Prentice-Hall, Englewood Cliffs, 2nd Edition,* 1992.**

[27] **R. E. Bellman, "Dynamic programming," *Princeton University Press, Princeton, New Jersey,* 1957.**

[28] **M. Qingming and P. Steenkiste, "On path selection for traffic with bandwidth guarantees," in *Network Protocols, 1997. Proceedings., 1997 International Conference on*, 1997.**

[29] **G. Apostolopoulos and S. K. Tripathi, "On reducing the processing cost of on-demand QoS path computation," in *Network Protocols, 1998. Proceedings. Sixth International Conference on*, 1998, pp. 80-89.**

[30] **A. Shaikh, J. Rexford, and K. G. Shin, "Evaluating the overheads of source-directed quality-of-service routing," in *Network Protocols, 1998. Proceedings. Sixth International Conference on,* 1998, pp. 42-51.**

[31] **A. Iwata and N. Fujita, "A hierarchical multilayer QoS routing system with dynamic SLA management," *Selected Areas in Communications, IEEE Journal on,* vol. 18, pp. 2603-2616, 2000.**

[32] **A. Forum, "network network interface (PNNI), v1.0 specification," 1996.**

[33] **D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, and J. McMaus, "Requirements for Traffic Engineering over MPLS," *Internet Draft,* 1998.**

[34] **Braden, "Integrated Services in the Internet Architecture: an Overview," *IETF RFC 1633,* June 1994.**

[35] **D. B. S. Blake, M. Carlson,E. Davies, Z.Wang andW. Weiss, "An Architecture for Differentiated Services," *IETF RFC 2475,* December 1998.**

[36] **A. Orda and A. Sprintson, "QoS routing: the precomputation perspective," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, 2000, pp. 128-136 vol.1.**

[37] **Z. Wang, "Intrenet QoS Architectures and Mechanisms for Quality of Service," *Lucent Technologies,* 2001.**

[38] **S. Harnedy, "The MPLS Primer: An Introduction to Multiprotocol Label Switching," *Upper Saddle River, NJ: Prentics Hall,* 2002.**

[39] **K. K. Nguyen and B. Jaumard, "A distributed and scalable MPLS architecture for next generation routers," in *High Performance Switching and Routing, 2008. HSPR 2008. International Conference on*, 2008, pp. 63-68.**

[40] **"Differentiated Services (diffserv) Charter," <*http://www.ietf.org/html.charters/diffservcharter.html*>.**

[41] **F. B. J. Heinanen, W.Weiss, and J. Wroclawski, "Assured Forwarding PHB Group," *Request for Comments RFC 2297, Internet Engineering Task Force,* 1999.**

[42] **K. N. V. Jacobson, and K. Poduri, "An Expedited Forwarding PHB," *Request for Comments RFC 2298, Internet Engineering Task Force,* 1999.**

[43] **Z. Zhang and e. al, "Decoupling QoS Control from Core Routers: A Novel Bandwidth Broker Architecture for Scalable Support of Guaranteed Services," *ACM SIGCOMM,* 2000.**

[44] **V. J. K. Nichols, and L. Zhang, "Two-bit Differentiated Services Architecture for the Internet," *Request for Comments RFC 2638, Internet Engineering Task Force,* 1999.**

[45] **F. B. Y. Bernet, P. Ford, R. Yavatkar, and L. Zhang, "A Framework for End-to-End QoS Combining RSVP/Intserv and Differentiated Services," *Internet-Draft <draftbernet-intdiff-00.txt>,* 2000.**

[46] **S. Nelakuditi, "Localized approach to providing quality-of-service," *Dept. of Computer Science and Engineering. Minneapolis: University of Minnesota,* p. 174, 2001.**

[47] **A. S. Alzahrani and M. E. Woodward, "End-to-end delay in localized QoS routing," in *Communication Systems, 2008. ICCS 2008. 11th IEEE Singapore International Conference on*, 2008, pp. 1700-1706.**

[48] **S. H. Alabbad and M. E. Woodward, "Localised credit based QoS routing," *Communications, IEE Proceedings-*, vol. 153, pp. 787-796, 2006.**

[49] **S. Nelakuditi, Z. Zhi-Li, R. P. Tsang, and D. H. C. Du, "On selection of candidate paths for proportional routing," *Computer Networks,* vol. 44, pp. 79-102, 2004.**

[50] **Y. Xin and A. Saifee, "Path selection methods for localized quality of service routing," in *Computer Communications and Networks, 2001. Proceedings. Tenth International Conference on*, 2001, pp. 102-107.**

[51] **D. Mitra and J. B. Seery, "Comparative evaluations of randomized and dynamic routing strategies for circuit-switched networks," *Communications, IEEE Transactions on*, vol. 39, pp. 102-116, 1991.**

[52] **R. J. Gibbens, F. P. Kelly, and P. B. Key, "Dynamic Alternative Routing: modelling and behaviour," *Teletraffic Science, Proceedings of 12th International Teletraffic Congress*, pp. 1019–1025, 1988.**

[53] **R. R. Stacey and D. J. Songhurst, "Dynamic Alternative Routing in the British Telecom trunk network," *International Switching Symposium, Phoenix*, 1987.**

[54] **A. Shaikh, J. Rexford, and K. G. Shin, "Evaluating the impact of stale link state on quality-of-service routing," *Networking, IEEE/ACM Transactions on*, vol. 9, pp. 162-176, 2001.**

[55] **R. J. Gibbens, F. P. Kelly, and P. B. Key, "Dynamic Alternative Routing," *Routing in Communication Networks, Prentice Hall*, 1995.**

[56] **J. Banks, J. S. Carson, and B. L. Nelson, "Discrete-Event System Simulation," *Prentice Hall*, 1996.**

[57] **V. Hlupic, "Simulation software: an Operational Research Society survey of academic and industrial users," in *Simulation Conference Proceedings, 2000. Winter*, 2000, pp. 1676-1683 vol.2.**

[58] **P. Deitel and H. Deitel, " Java how to program," *Pearson Education*, 2009.**

[59] **A. R. M. Shariff, "Some New Distributed Routing Models and Algorithms For Quality of Service Routing "** ***Computer Communication Networks,*** **2007.**

[60] **J. M. Pitts and J. A. Schormans, "Introduction to IP and ATM Design and Performance with Applications Analysis Software,"** ***John Wiley & Sons,*** **2002.**

[61] **C. Casetti, R. Lo Cigno, M. Mellia, M. Munafo, and Z. Zsoka, "A realistic model to evaluate routing algorithms in the Internet," in** ***Global Telecommunications Conference, 2001. GLOBECOM '01. IEEE,*** **2001, pp. 1882-1885 vol.3.**

[62] **R. Guerin and V. Peris, "Quality-of-Service in Packet Networks: Basic Mechanisms and Directions,"** ***Computer Networks Journal,*** **vol. 31, pp. 169-189, 1999.**

[63] **Y. Cui-Qing and A. V. S. Reddy, "A taxonomy for congestion control algorithms in packet switching networks,"** ***Network, IEEE,*** **vol. 9, pp. 34-45, 1995.**

[64] **P. Gevros, J. Crowcroft, P. Kirstein, and S. Bhatti, "Congestion control mechanisms and the best effort service model,"** ***Network, IEEE,*** **vol. 15, pp. 16-26, 2001.**

[65] **R. N. E. Crawley, B. Rajagopalan, and H. Sandick, "A Framework for QoS-based Routing in the Internet,"** ***RFC 2386,*** **1998.**

[66] **E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol Label Switching Architecture,"** ***Internet Draft*** **1998.**

[67] **F. A. Kuipers and P. Van Mieghem, "The impact of correlated link weights on QoS routing," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 2003, pp. 1425-1434 vol.2.**

[68] **L. Z. R. Braden, S. Berson, S. Herzog and S. Jamin, "Resource ReSerVation Protocol (RVSP) - Version 1 Functional Specification," *IETF RFC 2205*, Sept 1997.**

[69] **V. Paxson and S. Floyd, "Why we don't know how to simulate the internet," *Proceedings of the 1997 Winter Simulation Conference*, December 1997.**

[70] **E. W. Zegura, K. L. Calvert, and M. J. Donahoo, "A quantitative comparison of graph-based models for Internet topology," *IEEE/ACM Trans. Netw.*, vol. 5, pp. 770-783, 1997.**

[71] **M. Doar and I. Leslie, "How bad is naive multicast routing?," in *INFOCOM '93. Proceedings.Twelfth Annual Joint Conference of the IEEE Computer and Communications Societies. Networking: Foundation for the Future. IEEE*, 1993, pp. 82-89 vol.1.**

[72] **M. B. Doar, "A better model for generating test networks," in *Global Telecommunications Conference, 1996. GLOBECOM '96. 'Communications: The Key to Global Prosperity*, 1996, pp. 86-93.**

[73] **N. Spring, R. Mahajan, D. Wetherall, and T. Anderson, "Measuring ISP topologies with Rocketfuel," *Networking, IEEE/ACM Transactions on*, vol. 12, pp. 2-16, 2004.**

[74] **O. Heckmann, M. Piringer, J. Schmitt, and R. Steinmetz, "On realistic network topologies for simulation," in *Proceedings of the ACM SIGCOMM workshop on Models, methods and tools for reproducible network research* Karlsruhe, Germany: ACM, 2003.**

[75] **S. Chen and K. Nahrstedt, "Distributed Quality-of-Service Routing in High-Speed Networks Based on Selective Probing," in *Proceedings of the 23rd Annual IEEE Conference on Local Computer Networks*: IEEE Computer Society, 1998.**

[76] **C. Shigang and K. Nahrstedt, "On finding multi-constrained paths," in *Communications, 1998. ICC 98. Conference Record.1998 IEEE International Conference on*, 1998, pp. 874-879 vol.2.**

[77] **D. E. Comer, "Internetworking with TCP/IP: Principles, Protocols and Architecture," *Prentice Hall*, 2000.**

[78] **A. H. Mohammad and M. E. Woodward, "Localized Quality based QoS routing," in *Performance Evaluation of Computer and Telecommunication Systems, 2008. SPECTS 2008. International Symposium on*, 2008, pp. 209-216.**

[79] **A. S. Alzahrani and M. E. Woodward, "Localized quality-of-service routing using delay," in *Internet, 2008. ICI 2008. 4th IEEE/IFIP International Conference on*, 2008, pp. 1-6.**

[80] **A. S. Alzahrani and M. E. Woodward, "Residual bandwidth as localized QoS routing metric," in *Software, Telecommunications and Computer Networks, 2008. SoftCOM 2008. 16th International Conference on*, 2008, pp. 125-129.**

[81] **A. Forum, "Private Network-Network Interface Specification Version 1.1 (PNNI 1.1)," 2002.**

[82] **E. W. Zegura, K. L. Calvert, and M. J. Donahoo, "A quantitative comparison of graph-based models for Internet topology," *Networking, IEEE/ACM Transactions on,* vol. 5, pp. 770-783, 1997.**

[83] **E. W. Zegura, K. L. Calvert, and S. Bhattacharjee, "How to model an internetwork," in *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, 1996.**

[84] **G. Apostolopoulos, D. Williams, S. Kamat, R. Guerin, A. Orda, and T.Przygienda, "QoS Routing Mechanism and OSPF Extensions," *RFC 2676,*1999.**

[85] **J. Heidemann, K. Mills, and S. Kumar, "Expanding confidence in network simulations," *Network, IEEE,* vol. 15, pp. 58-63, 2001.**

[86] **S. H. Alabad, " Quality of Service Routing in Communication Networks: Algorithms and Partitioning Techniques," University of Bradford, 2006.**

[87] **A. A. Shaikh, "Efficient Dynamic Routing In Wide-Area Networks," *in Dept. of Computer Science and Engineering. Ann Arbor: The University of Michigan,* 1999.**

[88] **A. Feldmann, "Characteristics of TCP connections arrivals," in *SelfsimilarNetwork Traffic and Performance Evaluation,*" pp. 367--399, 2000.**

[89] **A. Feldmann, "Impact of non-poisson arrival sequences for call admission algorithms with and without delay," *GLOBECOM'96. IEEE Global Telecommunications Conference,* 1996.**

[90] **A. Feldmann, "On line Call Admission for High Speed Networks," *in School of Computer Science,* 1995.**

[91] **B. Peng, A. H. Kemp, and S. Boussakta, "Impact of network conditions on QoS routing algorithms," in *Consumer Communications and Networking Conference, 2006. CCNC 2006. 3rd IEEE,* 2006, pp. 25-29.**

[92] **C. Chekuri, S. Khanna, and F. B. Shepherd, "Edge-disjoint paths in planar graphs with constant congestion " *In Proceedings of the 38th ACM STOC* 2006.**

[93] **C. Chekuri and S. Khanna, "Edge Disjoint Paths Revisited " *In Proceedings of the 14th ACM-SIAM Symposium on Discrete Algorithms* 2003.**

[94] **A. Baveja and A. Srinivasan, "Approximation algorithms for disjoint paths and related routing and packing problems," *Mathematics of Operations Research* 2000.**

[95] **P. Jung-Heum, K. Hee-Chul, and L. Hyeong-Seok, "Many-to-Many Disjoint Path Covers in the Presence of Faulty Elements," *Computers, IEEE Transactions on,* vol. 58, pp. 528-540, 2009.**

[96] **D. Torrieri, "Algorithms for finding an optimal set of short disjoint paths in a communication network," *Communications, IEEE Transactions on,* vol. 40, pp. 1698-1702, 1992.**

[97] **R. G. Ogier, V. Rutenburg, and N. Shacham, "Distributed algorithms for computing shortest pairs of disjoint paths," *Information Theory, IEEE Transactions on,* vol. 39, pp. 443-455, 1993.**

[98] **F. A. Kuipers, "Quality of service routing in the internet : theory, complexity and algorithms," *Delft University Press,* 2004.**

[99] **A. Orda and A. Sprintson, "Efficient algorithms for computing disjoint QoS paths," in *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies,* 2004, p. 738.**

[100] **C. Peng and H. Shen, "A New Approximation Algorithm for Computing 2-Restricted Disjoint Paths*This research is supported by "Fostering Talent in Emergent Research Fields" program in Special Coordination Funds for promoting Science and Technology by Ministry of Education, Culture, Sports, Science and Technology," *IEICE - Trans. Inf. Syst.,* vol. E90-D, pp. 465-472, 2007.**

[101] **V. Guruswami, S. Khanna, R. Rajaraman, B. Shepherd, and M. Yannakakis, "Near-optimal hardness results and approximation algorithms for edge-disjoint paths and related problems," *J. Comput. Syst. Sci.,* vol. 67, pp. 473-496, 2003.**

[102] **L. Kleinrock and F. Kamoun, "Hierarchical Routing for large networks performance evaluation and optimization**

" *Computer Networks,* vol. 1, pp. 155-174, 1977.

[103] **J. Bhhrens and J. J. Garcia-Luna-Aceves, "Hierarchical Routing Using Link Vectors," *IEEE Infocom'98,* 1998.**

[104] W. C. Lee, "Topology aggregation for hierarchical routing in ATM networks," *ACM SIGCOMM Compuer Communication Review*, vol. 25, pp. 82-92, 1995.

[105] M. Steenstrup, "*Routing in Communications Networks*," *Englewood Cliffs, New Jersey: Prentice-Hall*, 1995.

[106] R. Izmailov, B. S. A. Iwata, and H. Suzuki, "PNNI Routing Algorithms for Multimedia ATM Internet," *NEC Research and Development*, vol. 38, pp. 60-73, 1997.

[107] B. Awerbuch, B. K. Y. Du, and Y. Shavitt, "Routing Through Teranode Networks with Topology Aggregation," *IEEE ISCC'98*, 1998.

[108] F. Hao and E. W. Zegura, "Scalability Techniques in QoS Routing," *College of Computing Georgia Institute of Technology*, 1999.

[109] L. Kleinrock and F. Kamoun, "Stochastic Performance Evaluation of Hierarchical Routing for large networks " *Computer Networks*, vol. 3, pp. 337-353, 1979.

[110] L. M. Breslau, "Adaptive source routing of real-time traffic in integrated services networks " *Univ. of Southern Califronia, Ph.D thesis*, 1995.

[111] L. Breslau, D. Estrin, and L. Zhang, "A simulation study of adaptive source routing in integrated services networks," *Computer Science Department, University of Southern California*, 1993.

[112] A. Orda and R. Guerin, "QoS-Based Routing in Networks with Inaccurate Information: Theory and Algorithms," *IEEE/ACM Transactions on Networking,* vol. 7, pp. 605-614, 1999.

[113] G. Apostolopoulos, R. Guerin, and S. Kamat, "Implementation and performance measurements of QoS routing extensions to OSPF," *Proceedings of IEEE INFOCOM,* 1999.

[114] M. Peyravian and A. D. Kshemkalyani, ""Network path caching: Issues, algorithms and a simulation study," *Computer Communications Journal,* vol. 20, pp. 605-614, 1997.